University of Alberta

DESIGN OF A DYNAMIC SEMICONDUCTOR FILE MEMORY WITH SINGLE-TRANSISTOR SENSE AMPLIFIERS

by

Kristopher C. Breen ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**

Department of Electrical and Computer Engineering

Edmonton, Alberta
Spring 2005

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-22423-6
Our file Notre référence
ISBN: 978-0-494-22423-6

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

This thesis introduces a new architecture and new techniques for semiconductor file memory design, based on dynamic random access memory (DRAM). The ideas presented aim to create a less expensive, lower performance memory than DRAM. Such a memory can improve the performance of a computer memory system, or lower system cost for equivalent performance. The proposed architecture is founded on the concept of single-transistor sense amplifiers. These amplifiers require much less area than conventional DRAM amplifiers, permitting higher density. To support single-transistor amplifiers, several techniques are employed, including time-multiplexed sensing, a two-pass write operation, a two-step refresh operation and aged references. Analysis using 0.13-μm technology reveals that worst case noise margins are satisfied using 80-cell bitlines, though device mismatch is problematic. Compared to conventional DRAM, density is increased by 17%. Latency is three orders of magnitude greater, while throughput is comparable for block access.

Acknowledgements

This research was funded by scholarships from the Natural Sciences and Engineering Research Council of Canada (NSERC), the Alberta Informatics Circle of Research Excellence (iCORE), and the University of Alberta. Additional funding and support for research equipment was provided by Micronet R&D and the Canadian Microelectronics Corporation.

I thank my supervisor, Dr. Duncan Elliott, for being very supportive during my studies and for sharing many interesting ideas with me. I'd also like to thank Dr. Bruce Cockburn and Dr. Vincent Gaudet for their guidance, and Paul Greidanus for being very responsive to my problems and concerns relating to CAD tools. The experience of sharing ideas with fellow students has been tremendously helpful, and I'd like to thank Tyler Brandon, Christian Giasson, Craig Joly, John Koob, Kaston Leung, Nitin Parimi, Kamlesh Raiter and Sue Ann Ung for their assistance with my research, as well as Amir Alimohammad, Edmund Fung, Jesús Hernández Tapia, Dan Leder, Madhura Purnaprajna, Ashwin Rao and Mimi Yiu for contributing to a supportive work environment in the VLSI Design Lab.

To Janna

Table of Contents

1 Introduction **1**
- 1.1 Overview . . . 1
- 1.2 Motivation . . . 1
- 1.3 Applications . . . 2
  - 1.3.1 Computer Memory System Enhancement . . . 2
  - 1.3.2 Portable Solid-State Disk . . . 3
  - 1.3.3 Low-Latency Web Server . . . 3
- 1.4 Thesis Organization . . . 3

2 Background and Prior Work **5**
- 2.1 Overview . . . 5
- 2.2 Semiconductor Memories . . . 5
- 2.3 DRAM . . . 7
  - 2.3.1 Basic Concept . . . 7
  - 2.3.2 Memory Cell Operation . . . 8
  - 2.3.3 Organization and Architecture . . . 9
  - 2.3.4 Sensing Techniques . . . 12
  - 2.3.5 Noise in DRAMs . . . 15
  - 2.3.6 Leakage in Sub-0.13-μm DRAMs . . . 16
  - 2.3.7 DRAM Manufacturing Process Considerations . . . 17
  - 2.3.8 Cost and Economics . . . 19
- 2.4 File Memory . . . 20
  - 2.4.1 File Memory in a Memory Hierarchy . . . 20
  - 2.4.2 Prior File Memory Implementations . . . 21
- 2.5 Summary . . . 25

3 Proposed Dynamic File Memory Architecture **27**
- 3.1 Overview . . . 27
- 3.2 Introduction to the Proposed Architecture . . . 28
  - 3.2.1 Concept . . . 28
  - 3.2.2 Requirements . . . 29
  - 3.2.3 Basic Operation . . . 31
- 3.3 Description of the Proposed Architecture . . . 32
  - 3.3.1 Array Architecture . . . 33

3.3.2 Sensing Scheme 34
3.3.3 Core Architecture 42
3.3.4 Memory Architecture 45
3.3.5 Reference Scheme 47
3.3.6 Data Bus Amplifiers 51
3.3.7 Data Converters 54
3.3.8 SRAM Buffer 55
3.4 Summary 55

4 Analysis and Simulation Results 57
4.1 Overview 57
4.2 Method 57
4.3 Density Improvement Analysis 59
4.4 Functional Simulation 62
4.5 Sense Amplifier Simulation 64
4.6 Bitline Biasing 64
4.7 Noise Margins 67
4.7.1 Parasitic Capacitive Coupling Noise 67
4.7.2 Process Variations and Offset Noise 70
4.7.3 Effect of Leakage on Noise Margins 72
4.7.4 Overall Noise Margins 74
4.8 Performance 77
4.9 Power Consumption 80
4.10 Examination of the Wine Cellar Technique 81
4.11 Primary Amplifier Circuit Evaluation 84
4.12 Summary 86

5 Multilevel Operation 89
5.1 Overview 89
5.2 Area Analysis 90
5.3 Functional Simulation 91
5.4 Noise Margins 91
5.5 Performance and Power Consumption 93
5.6 Multilevel Wine Cellar Technique 95
5.7 Summary 98

6 Conclusion 99
6.1 Synopsis 99
6.2 Architecture 99
6.3 Results 100
6.4 Accomplishments 101
6.5 Challenges 102
6.6 Future Work 103
6.7 Summary 105

Bibliography **106**

A Further Discussion **113**
A.1 Density, Yield and Cost 113
A.2 Coupling Noise Cancellation 114
A.3 Page Load Times 116

B Array Model and Simulation Schematics **119**
B.1 Array Model 119
B.2 Simulation Schematics 121

C Details of Analytical Analyses **131**
C.1 Performance Calculations 131
C.2 Tables of Calculated and Raw Data 134
C.3 Matlab Code For Wine Cellar Technique Analysis 146

D Simulation Stimulus Scripts and Waveforms **151**

E Verilog-A HDL Code **187**

List of Tables

4.1 Area of the Proposed Architecture for Different Bitline Lengths . . 62
4.2 Capacitance and Bias Voltage for Different Bitline Lengths 66
4.3 Core Performance Results 78

6.1 Important Characteristics 101

B.1 Array Parameters 120

List of Figures

2.1 Semiconductor Memory Classification 6
2.2 1T1C Unit Storage Cell 7
2.3 Positive Feedback Differential Sense Amplifier 13
2.4 Time Multiplexed Sensing Structure 14
2.5 Common Computer Memory Hierarchy 22
2.6 NAND Structured Memory Cells 24

3.1 Conceptual Diagram of Sense Amplifier Transistors Along a Data Bus 32
3.2 Four Potential Read Amplifier Configurations 36
3.3 Four-Transistor Sense Amplifier Configuration 37
3.4 Alternate Sense Amplifier Configuration 38
3.5 Approximate I/O Characteristic for the NMOS 4T Sense Amplifier . 39
3.6 Timing Diagram Showing Write Coupling Problem and Solution . . 41
3.7 Sense Amplifier Organization Scheme 43
3.8 Signal Routing in a Sub-Array Unit 44
3.9 Block Diagram 46
3.10 Signal Margins for Conventional References and Wine Cellar Technique 49
3.11 Wine Cellar Technique with Multilevel Memory 50
3.12 Current Mirror Bus Amplifier 53
3.13 Bus Precharge Amplifier 53

4.1 Area Comparison of Two Memories with 2^{30} Dynamic Cells 61
4.2 Functional Simulation Results 63
4.3 Sense Amplifier I/O Characteristics for Typical, Fast and Slow Models 64
4.4 Worst Case Noise due to Capacitive Coupling for Different Bitline Sizes 69
4.5 Noise Margins Remaining After Capacitive Coupling for Different Bitline Sizes 70
4.6 Noise due to Process Variations For Different Bitline Sizes 71
4.7 Signal Degradation due to Read/Write Leakage for Different Sub-Wordline Lengths 74
4.8 Maximum Sub-Wordline Length for 1% Signal Degradation During Read and Write 75

4.9 Worst Case Noise Margins for Different Bitline Lengths 75
4.10 Read Time for Different Data Bus Amplifier Input Resistances . . . 79
4.11 Sense Amplifier Current Output Versus Cell Storage Time 81
4.12 Comparison of Conventional and WCT Schemes: Typical Noise Margin 82
4.13 Cell Voltage and 10% Thresholds Versus Cell Storage Time 83
4.14 Comparison of Conventional and WCT Schemes: Read Errors . . . 84
4.15 Current Mirror Amplifier Output Current Characteristic 85
4.16 Current Mirror Amplifier Transient Characteristic 85
4.17 Bus Precharge Amplifier Output Voltage Characteristic 86
4.18 Bus Precharge Amplifier Transient Characteristic 86

5.1 Area Per Bit Versus Number of Levels Per Cell 91
5.2 Functional Simulation Results for Multilevel Storage 92
5.3 Worst Case Signal and Noise Strengths for 64-cell Bitlines and 25-fF Cells 94
5.4 Sense Amplifier Output Current Transient Characteristic for Four-Level Storage 95
5.5 Comparison of Conventional and WCT Noise Margins for Four-Level Storage (for a Stored '11') 96
5.6 Cell Voltage and 10% Thresholds for Four-Level Storage (for a Stored '11') 97
5.7 Comparison of Conventional and WCT Percentage of Cells in Error for Four-Level Storage 97

A.1 Page Load Time for Various Page Sizes 117

B.1 1T NMOS CSTA Schematic 121
B.2 1T NMOS CDTA Schematic 122
B.3 1T PMOS CSTA Schematic 122
B.4 1T PMOS CDTA Schematic 123
B.5 Bitline Capacitance Measurement Schematic 124
B.6 Memory Cell Schematic 124
B.7 Memory Array Schematic – 32 x 5 125
B.8 Memory Array Schematic – Closeup 126
B.9 Sense Amplifier Schematic 126
B.10 Data Bus Model Schematic 127
B.11 Core Testbench Schematic 127
B.12 Current Mirror Bus Amplifier Test Schematic 128
B.13 Bus Precharge Bus Amplifier Test Schematic 129
B.14 Opamp Bus Amplifier Schematic 129

C.1 Two Level Area Analysis Data 134
C.2 Bitline Capacitance and Optimal Bias Point Analysis Data 137
C.3 Bitline Coupling Noise Analysis Data (Page 1) 138

C.4 Bitline Coupling Noise Analysis Data (Page 2) 139
C.5 Process Variation Analysis Data 140
C.6 Read/Write Leakage Analysis Data 141
C.7 Total Noise Margin Analysis Data 142
C.8 Performance, Refresh, Data Converter and SRAM Analysis Data . . 143
C.9 Power Analysis Data 144
C.10 Wine Cellar Technique Analysis Data 145

D.1 Functional Simulation Stimulus Waveforms 153
D.2 Coupling Measurement Stimulus Waveforms – No Coupling 157
D.3 Coupling Measurement Stimulus Waveforms – Worst Case '0' . . . 160
D.4 Coupling Measurement Stimulus Waveforms – Worst Case '1' . . . 163
D.5 Performance Measurement Stimulus Waveforms 166
D.6 Bitline Power Measurement Stimulus Waveforms 169
D.7 Core Power Measurement Stimulus Waveforms 172
D.8 Wine Cellar Technique Simulation Stimulus Waveforms 175
D.9 Current Mirror Amplifier Transient Simulation Stimulus Waveforms 177
D.10 Bus Precharge Amplifier I/O Simulation Stimulus Waveforms . . . 179
D.11 Bus Precharge Amplifier Transient Simulation Stimulus Waveforms 181
D.12 Multilevel Functional Simulation Stimulus Waveforms - Page 1 . . . 184
D.13 Multilevel Functional Simulation Stimulus Waveforms - Page 2 . . . 185

E.1 Top Level Interconnection Schematic for Functional Simulation . . 192
E.2 Top Level Interconnection Schematic for Multilevel Functional Simulation 198

Nomenclature

List of Acronyms

1T1C	One Transistor, One Capacitor, page 7
ADC	Analog-to-Digital Converter, page 47
BBM	Bad Block Marking, page 24
BL	Bitline, page 14
BTBT	Band-to-band tunneling, page 16
CDTA	Common Drain Transconductance Amplifier, page 35
CMOS	Complimentary Metal Oxide Semiconductor, page 17
CSR	Column Select Read, page 37
CSTA	Common Source Transconductance Amplifier, page 35
CSW	Column Select Write, page 41
DAC	Digital-to-Analog Converter, page 47
DRAM	Dynamic Random-Access Memory, page 2
ECC	Error-Correcting Code, page 24
EEPROM	Electrically Erasable Programmable Read-Only Memory, page 6
EPROM	Erasable Programmable Read-Only Memory, page 6
ESDC	Extended Storage Disk Cache, page 21
FeRAM	Ferroelectric Random-Access Memory, page 6
GWL	Global Wordline, page 44
I/O	Input/Output, page 9

LRD	Local Row Decode, page 44
MLDRAM	Multilevel DRAM, page 22
NMOS	N-type Metal Oxide Semiconductor, page 35
PC	Personal Computer, page 3
PMOS	P-type Metal Oxide Semiconductor, page 35
ROM	Read-Only Memory, page 6
SAS	Sub-Array Select, page 44
SRAM	Static Random-Access Memory, page 3
WCT	Wine Cellar Technique, page 82
WL	Wordline, page 14

List of Symbols

γ	Refresh-busy ratio, page 127
λ	Average number of faults per chip, page 108
A	Chip area for yield calculations, page 108
A_{chip}	Area of a memory chip, page 60
A_{core_unit}	Area of a core unit (an array with supporting sense amplifier and local row decode strips), page 59
A_{dram}	Area of DRAM, page 61
A_{fm}	Area of file memory, page 61
$A_{periphery}$	Chip area occupied by peripheral circuitry, page 60
$A_{sense_{dram}}$	Area of sense amplifiers in DRAM, page 61
$A_{sense_{fm}}$	Area of sense amplifiers in file memory, page 61
$A_{sub-array}$	Area of a sub-array, page 59
C_b	Bitline capacitance, page 9
C_s	Cell capacitance, page 9
d	Average defect density, page 108
F	DRAM $\frac{1}{2}$ pitch, page 10
H_{lrd}	Height (along the wordlines) of a local row decoder, page 59
i_L	Total leakage current, page 73
i_{Lpn}	PN junction leakage current, page 73
i_{Lsub}	Subthreshold leakage current, page 73
I_l	Leakage current, page 15
N	Number of stored voltage levels (2 for standard DRAM), page 15
N_{banks}	Number of banks in a memory chip, page 60
N_b	Number of bitlines in a sub-array, page 59
N_{cbl}	Number of cells per bitline, page 73

$N_{core\_units/bank}$	Number of core units per memory bank, page 60
N_w	Number of wordlines in a sub-array, page 59
P_b	Bitline pitch, page 59
P_w	Wordline pitch, page 59
PRE	Precharge signal, page 44
Q_c	Soft-error critical charge, page 15
$V_{bias_{opt}}$	Optimal bias voltage, page 66
V_{bias}	Bias voltage, page 52
$v_{bl\_leakage}$	Noise voltage due to floating bitline leakage, page 75
V_{bl}	Bitline voltage, page 73
$v_{cell\_leakage}$	Worst case signal degradation allowed by cell leakage between refreshes, page 75
V_{cell}	Cell voltage, page 9
$v_{coupling}$	Coupling noise voltage, page 75
$v_{critical}$	Voltage due to soft-error critical charge after charge sharing, page 15
V_{DDP}	Boost voltage, page 114
V_{DD}	Supply Voltage, page 8
v_{gpv}	Global process variation noise voltage, page 75
$v_{leakage}$	Voltage reduction due to cell leakage after charge sharing, page 15
V_{lower_bound}	Amplifier input lower bound, page 66
v_{nm}	Noise margin on a bitline, page 15
v_{noise}	Noise voltage, page 15
V_{pre}	Bitline precharge voltage, page 9
V_{ref_mid}	Midpoint of reference voltages, page 50
v_{signal}	Maximum signal voltage after charge sharing, page 15

V_{th}	Threshold voltage, page 35
W_{sa}	Width (along the bitlines) of a sense amplifier, page 59
Y	Chip yield, page 108
$\Delta t_{r_{max}}$	Time between refresh cycles, page 15

List of Terms

cell ratio	The ratio of bitline capacitance to cell capacitance in a dynamic memory array, page 9
charge-transfer ratio	The ratio of cell capacitance to the combined capacitance of a cell and bitline in a DRAM array, page 9
column	The group of cells connected to a bitline, in the "Y" dimension, page 10
crosspoint array	Array organization where a memory cell resides at every intersection of a wordline and bitline, page 10
DRAM $\frac{1}{2}$ pitch	One half of the minimum bitline or wordline pitch, page 10
file memory	A class of computer memory that is slower and cheaper than main memory but faster than mass storage such as magnetic disk, page 1
folded bitline	Array organization where each pair of adjacent bitlines is connected to a single sense amplifier, page 10
global process variation	Variation of device characteristics where all devices on a die deviate from typical characteristics, page 70
local process variation	Variation of device characteristics where the characteristics of each device within a spatial region on a die are different, page 70
memory core	The part of a memory that contains of the memory cells, sense amplifiers, local decoders, and other circuitry that is repeated for each sub-array, page 9
memory hierarchy	An organizational system for computer memory, in which each level of memory is smaller, faster, and more expensive than the one below it, page 21
multi-division	The partitioning of bitlines and wordlines among sub-arrays in the memory core, page 42
noise margin	The maximum amount of noise that can be tolerated in the presence of a data signal without an error occurring, page 15
non-volatile memory	Memory that retains data in the absence of supplied power, page 6

open bitline	Array organization where each bitline is independent of its neighbours and is connected to a separate sense amplifier, page 10
periphery	The part of a memory that contains primary control, I/O, and other circuitry that is not repeated with each sub-array, page 9
reference bitline	An extra bitline within a sub-array whose cells store reference values instead of data values, page 47
refresh-busy ratio	The proportion of operating time spent refreshing memory cells, page 79
row	The group of cells connected to a wordline, in the "X" dimension, page 10
sense amplifier	A circuit that connects to a bitline or bitline pair, and is repeated at an integer multiple of the bitline period; responsible for performing read, write, and refresh operations on a bitline, page 12
sub-array	A unit of memory cells that is repeated many times within the memory core to form the storage area of the memory, page 11
sub-wordline	A wordline segment that exists in a sub-array, page 11
two-pass write	A technique where an entire sub-row is written twice to reduce coupling noise during the write operation, page 42
volatile memory	Memory that does not retain data in the absence of supplied power, page 6
Wine Cellar Technique	A reference scheme where references are stored in memory cells until data is to be read, and then these stored references are compared to stored data to reconstruct data values, page 49

Chapter 1

Introduction

1.1 Overview

This thesis introduces a new architecture and new techniques for semiconductor file memory design. File memory is a class of computer memory that is slower and cheaper than main memory, but faster than mass storage such as magnetic disk. By introducing file memory into a computer memory system, the performance of that system can be improved and/or the cost of that system can be reduced. This thesis explains how the new architecture and new techniques can be used to design semiconductor file memory, and it evaluates their feasibility and effectiveness.

This chapter presents the motivation for this work, along with a few basic concepts that contribute to the reader's understanding of the remainder of this dissertation. It then presents the intended application domain, and concludes with a guide to the dissertation's organization.

1.2 Motivation

Performance and cost are two extremely important metrics in the evaluation of a digital computer. The memory system of a digital computer plays a large role in determining both of these.

It is well known that the use of multiple layers of cache in a computer memory system allows very good performance, while allowing the memory system to be economical. However, the extent to which caching can improve on the performance and cost of a memory system depends on the availability of suitable memory technologies to serve as different levels in the cache hierarchy. There are technologies currently available that are ideal for use as cache between the processor and main memory. However, between main memory and disk, there have been none developed that are enough to merit widespread use. In light of the fact that there is a steadily increasing performance gap between main memory and disk [24], there is good reason to investigate memory technologies that can fill this gap.

The motivation for the work in this thesis is to create a technology that can serve as a cache layer between main memory and disk. As is discussed in detail in chapter 2, recent research has shown that the inclusion of a cache layer between main memory and disk can substantially improve system performance and/or reduce system cost [18]. The right technology for this cache layer could find extensive commercial adoption.

Dynamic memory technology, which is the basis of dynamic random-access memory (DRAM), presents an ideal starting point for the development of file memory. Existing DRAM architectures offer substantial opportunities for tradeoffs that reduce cost in exchange for other factors such as performance and power consumption. The fact that DRAM is already so widely used means that little change to existing production facilities and industrial paradigms would be required for a new DRAM-based memory. In short, with the right architecture, semiconductor file memory can be economically developed for use in computer memory hierarchies, and potentially for use in other applications as well.

1.3 Applications

There are currently several possible application domains for semiconductor file memory. This section presents a few of the primary, most likely applications.

1.3.1 Computer Memory System Enhancement

As discussed previously, the target application for semiconductor file memory in the context of this thesis is computer memory system enhancement. The most likely market for the technology is the commercial personal computer (PC) market. Semiconductor file memory technology can provide increased performance, reduced cost, or both to PC consumers. However, a low-cost high-density semiconductor memory could also be appealing in servers given the right economic conditions. In supercomputers, the concept of file memory has already been applied in the Cray Y-MP, but higher in the memory hierarchy. The Cray Y-MP used static random-access memory (SRAM) as main memory, and DRAM as a solid-state disk (SSD) for caching data between main memory and disk [24]. Semiconductor file memory could be applied in a similar way, providing a cache between the SSD and disk.

1.3.2 Portable Solid-State Disk

As miniaturization and portability of electronics both become increasingly popular, the market for portable solid-state disks is growing. Current technologies for this purpose primarily include flash memory and DRAM. Though the volatility of DRAM, as with a DRAM-based semiconductor file memory, presents some challenges in this domain, the low cost and high storage density that could be achieved could make such technology very desirable.

1.3.3 Low-Latency Web Server

Semiconductor file memory could be used in a low-latency and low-cost web server. Disk storage could be completely replaced by file memory. The web server would need to have a low enough capacity requirement to make this economical, but it would still have much greater capacity than if conventional DRAM were used.

1.4 Thesis Organization

This dissertation is organized into six chapters. Chapter one serves as this introduction. The design of file memory draws on two major areas of research, those being DRAM integrated circuit design and file memory integration into a real system. Chapter two provides a thorough background in both of these areas. Chapter three describes the specific implementation of the file memory architecture and techniques that are proposed in this dissertation. Numerous options for each aspect of the architecture are presented, along with a discussion of the tradeoffs associated with each option. After the options are discussed, some specific design choices are made for the purpose of thorough analysis and simulation. Chapter four contains the results from analysis and simulation of the proposed architecture. Although enough results are presented to verify that the architecture satisfies all of its requirements, sufficient results are presented to characterize the architecture to the extent that a designer working with it could make accurate decisions about the appropriateness of tradeoffs. Chapter five examines the opportunities and challenges associated with incorporating multilevel storage into the proposed architecture. Finally, chapter six summarizes the work that has been presented, and proposes future research directions that can be taken with this work and with similar work in the same area.

Chapter 2

Background and Prior Work

2.1 Overview

Computer memories have been the topic of extensive research for many decades. Countless innovations and systems have been created in an effort to improve metrics like performance, power consumption, reliability, and as in our case, cost.

The information contained in this chapter is a summary of the academic and industrial developments in DRAMs and file memories that are most relevant to the work of this thesis. It includes background information on various significant aspects of DRAM, such as core circuitry, architecture, organization, and manufacturing considerations. The principles of file memory are discussed, and a survey of prior file memory implementations is also presented.

2.2 Semiconductor Memories

Semiconductor memories can be generally divided into two broad categories based on their volatility, or ability to retain data in the absence of supplied power. Volatile memory can be sub-classified based on the method of data retention used, dynamic or static. Dynamic volatile memory, known as Dynamic Random Access Memory (DRAM), requires periodic refreshing to retain data, while static volatile memory, known as Static Random-Access Memory (SRAM), does not. Non-volatile memory can be sub-classified based on the writability of the memory: memory that can be written to only once versus memory that can be rewritten. One-time

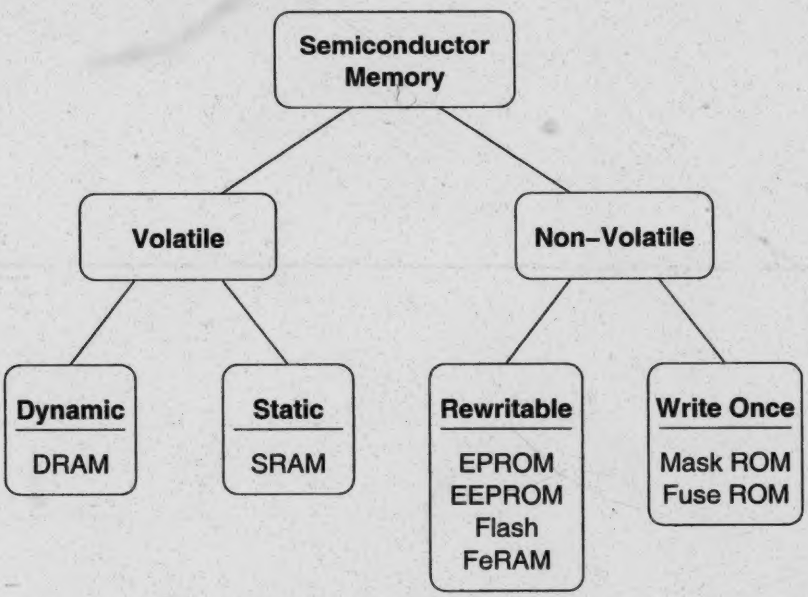


Figure 2.1: Semiconductor Memory Classification [14, 15, 27]

write memories include mask programmable and fuse programmable Read-Only Memory (ROM), and rewritable memories include Erasable Programmable Read-Only Memory (EPROM), Electrically Erasable Programmable Read-Only Memory (EEPROM), flash memory, and Ferroelectric Random-Access Memory (FeRAM), among others [27]. The classification of semiconductor memories can be seen diagrammatically in figure 2.1.

DRAM is the most dense, and therefore least expensive per bit, of all types of semiconductor memory that can be written during system operation. It offers good read and write performance, but consumes a fairly substantial amount of power compared with other memories. SRAM is more expensive per bit than DRAM, but offers the best performance of rewritable semiconductor memory.

Non-volatile memories, in general, have reasonably high density, and often have similar read performance to DRAM; however, most presently available rewritable non-volatile memories have poor write performance. Furthermore, they are only capable of a limited number of rewrites, which prevents them from competing with DRAM as an inexpensive, well-performing memory for computing systems.

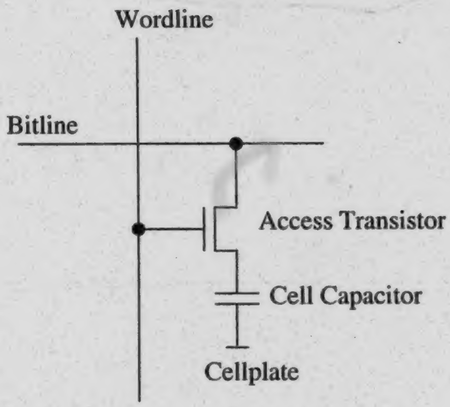


Figure 2.2: 1T1C Unit Storage Cell

A wealth of further information on the various types of semiconductor memories and their operation can be found in [14], [37] and [38].

2.3 DRAM

As mentioned previously, DRAMs are inexpensive in terms of cost per bit. At the same time, DRAMs offer relatively low latency access combined with a good throughput rate. These characteristics are the result of a design philosophy that focuses first on creating a memory that is as dense as possible, and then on encapsulating it in an architecture that maximizes performance.

2.3.1 Basic Concept

The salient characteristic of a modern DRAM is the use of the one transistor and one capacitor (1T1C or sometimes just 1T) unit storage cell. This cell, first introduced by Robert H. Dennard of IBM in 1968 [7], is shown schematically in figure 2.2.

The 1T1C cell can be designed in an extremely small area on an integrated circuit, allowing a large number of cells on a single chip, so that the resulting product has a low production cost per bit. The challenge in using the 1T1C cell is operating it in a large scale array with good performance. This requires an elaborate organization scheme that is described later in this section.

2.3.2 Memory Cell Operation

The 1T1C cell stores a bit by representing that bit as a charge stored across the cell capacitor. A positive charge represents a logic "high," while a negative charge represents a logic "low." The access transistor functions as a switch, controlled by the wordline, that connects the cell capacitor to the bitline.

The operation of the 1T1C cell is conceptually very simple. To store a bit in the cell, a "high" voltage level is applied to the wordline connected to the desired cell. At the same time, the voltage level to be stored is applied to the bitline that is connected to the desired cell. With the access transistor active, the cell capacitor is charged from the bitline with a "high" or "low" charge. The access transistor is then deactivated and the stored charge remains on the cell capacitor.

To read a bit from the cell, the capacitive bitline is left floating at a precharge voltage (usually $V_{DD}/2$), and the access transistor is activated. If a "high" voltage level is stored in the cell, then the bitline voltage will increase; if a "low" voltage level is stored, then the bitline voltage will decrease. Either way, a sense amplifier on the bitline detects the change in voltage and amplifies this voltage to a full logic "high" or "low" level. Because the access transistor is still open when this amplification occurs, the cell that was read is restored to its full original logic level. Once this restoration is complete, the access transistor is deactivated, and a new operation can begin.

The extent to which the bitline voltage increases or decreases during a read operation is determined by capacitive charge sharing. Both the bitline and memory cell have a fixed capacitance, with the bitline capacitance normally being five to ten times larger than that of the memory cell. When the access transistor is activated for a read operation, the charge on the cell capacitor is shared with the charge on the bitline to generate a change in bitline voltage given by equation 2.1:

$$\Delta V = (V_{cell} - V_{pre}) \frac{C_s}{C_b + C_s} , \tag{2.1}$$

where ΔV is the change in bitline voltage due to charge sharing, V_{cell} is the stored cell voltage, V_{pre} is the bitline precharge voltage, C_s is the cell capacitance, and C_b

is the bitline capacitance. The ratio $C_s/(C_b + C_s)$ is often referred to as the "charge-transfer ratio." Another ratio, C_b/C_s, is a useful indicator for a DRAM array, and is sometimes referred to as the "cell ratio."

Because the bitline capacitance is much larger than the cell capacitance, the ΔV value in equation 2.1 is normally relatively small. For that reason, the sense amplifier has to be very sensitive to small changes in voltage in order to adequately detect stored logic levels. Sense amplifiers and sensing techniques are examined more closely in section 2.3.4.

2.3.3 Organization and Architecture

For a DRAM containing millions of storage cells to have high density and good performance, the cells must be organized into an efficient pattern. Modern DRAM chips exhibit a very high degree of organization. At the highest level, DRAMs are partitioned into core and periphery regions. The core region consists of the memory cells along with supporting circuits that are repeated at a frequency equal to an integer multiple of that of the bitlines or wordlines. The periphery region consists of control circuitry, I/O pads, data buffers, synchronization circuitry, voltage conversion circuitry, and other circuits whose functions relate directly to the specific architecture in which they are employed. The core regions are sub-organized into an array region, which contains the memory cells themselves, and another region that contains sense amplifiers, hierarchical wordline drivers and bitline twist strips. The following subsections describe the organization of each DRAM region.

2.3.3.1 Array Organization

The memory array is a two-dimensional array of memory cells, with wordlines running parallel in one dimension (normally referred to as the "X" dimension) and bitlines running parallel in the other dimension (normally referred to as the "Y" dimension), such that wordlines and bitlines are perpendicular to each other. Due to this organization, the group of cells connected to a single wordline are often referred to as a "row," and the group of cells connected to a single bitline or a bitline pair are often referred to as a "column."

The most important aspect of array organization is the bitline structure. There are two predominant bitline structures in modern DRAMs. Those are the open bitline structure, originally introduced by Karl Stein et al. of Siemens in 1972 [39], and the folded bitline structure, introduced by Robert Harland of MOSAID Technologies in 1977 [11]. Every DRAM produced today uses one of these two bitline organizations, or else a direct variant or a hybrid of the two.

The open bitline organization, also referred to as "crosspoint" organization, is a simple scheme in which a memory cell resides at every intersection of a wordline and a bitline. In an open bitline structure, each bitline within an array is independent of its neighbours and is connected to a separate sense amplifier. The folded bitline organization, on the other hand, is a scheme in which a memory cell resides only at every second intersection between a wordline and a bitline[1]. Each pair of adjacent bitlines in a folded array is connected to a single sense amplifier.

The advantage of the open bitline structure is that it allows the tightest packing of memory cells possible. Using an open bitline array, a memory cell can be fit into a $6F^2$ area, where F is the DRAM $\frac{1}{2}$ pitch, defined as one half of the minimum bitline or wordline pitch[2]. DRAMs designed using an open bitline organization are more dense than those using a folded array organization, which features an $8F^2$ cell. However, folded arrays exhibit better signal to noise performance than open arrays when differential sensing is employed. This is because each pair of adjacent bitlines is connected to a single sense amplifier, with one bitline carrying the signal and the other serving as a reference. Since the bitlines are located next to one another, most noise generated from nearby sources in the array is common-mode, and is rejected by a differential sense amplifier. For this reason, almost all modern DRAM designs now use the folded array.

---

[1] In a physical implementation of a folded array it is not strictly necessary that every second intersection is a cell location, so long as one in two intersections on average is a cell location.

[2] F is sometimes also defined as the minimum feature size (wire width or spacing).

2.3.3.2 Core Organization

The memory core is organized hierarchically, with sub-arrays as the base elements. The capacitance of a bitline needs to be minimized so that a sufficiently strong signal is developed during read, as described by equation 2.1. Furthermore, the wordline and bitline capacitance both need to be minimized so as to minimize power consumption and improve performance. To accomplish this, the memory array is broken into sub-arrays that each have their own supporting circuitry. A typical sub-array in a 1-Gb DRAM with folded bitlines [19] has 256 cells per bitline and 512 cells per "sub-wordline", with a few additional of both types of line to provide static redundancy and to avoid photolithographic problems [16]. The term "sub-wordline" is used to describe wordline segments so as to distinguish them from global wordlines that run across multiple sub-arrays.

Mated with each sub-array is a block of sense amplifiers and a local row decoder block (assuming a hierarchical wordline scheme is used). Each of these sub-array units with their surrounding logic is repeated numerous times in two dimensions (for example, these units might be arranged in a 16x32 grid) to form a block of the core. Each core block will then have associated column decode, global row decode, and control logic involved in controlling data flow and moving data from the core to the periphery.

2.3.3.3 Chip Organization

At the highest level of organization within a DRAM, core blocks are grouped together (logically, but not necessarily physically) to form banks. For example, a DRAM might have four banks, with each bank comprising two core blocks, as in one DRAM developed by Samsung [19].

The physical organization of banks on a DRAM chip is done in such a way as to maximize parallelism and reduce clock skew. A number of novel organizations have been conceived, though a thorough discussion is outside the scope of this thesis. A demonstrative organization can be found in a paper by Sakashita et al. [33], and much discussion is contained in Sharma's recent text [38].

2.3.4 Sensing Techniques

One of the most defining aspects of a DRAM design is the sensing technique used to read (and restore) data from a memory cell. Many techniques and circuits have been developed since the advent of DRAM, and they can be broadly classified based on two attributes. The first is the amplifier circuit type, which can be single-ended (non-differential) or differential. The second is the operation mode of the amplifier, which can be voltage mode, current mode, or charge mode [10].

This section briefly presents the most important sensing techniques. In the discussion that follows, it is important to note that the term "sense amplifier" refers not just to an amplification circuit, but to all of the circuitry local to a bitline that is required to properly write, read, and refresh any cell on that bitline. This terminology is used throughout this entire thesis.

2.3.4.1 Single-Ended Sensing

Single-ended sense amplifiers are found in almost no modern memory designs because of the superior performance and noise rejection capabilities of their differential counterparts. However, they have the advantages of being simple, having the potential to occupy very little silicon area, and the potential to consume less power than a differential amplifier.

The earliest DRAMs, designed by Intel, used three transistor cells and employed single-ended sense amplifiers. The sense amplifier design consisted of four transistors: one for precharging the "write" bitline, a column enable transistor, a bias transistor, and a sensing transistor whose gate connected to the "read" bitline [30]. When a cell was selected and a signal was transferred to the "read" bitline, the sense transistor would drive a near zero current to an output pin to indicate a logic "high" or a 400-μA current to indicate a logic "low."

Today, single-ended sensing finds a few applications in non-volatile memories such as EEPROMs [26]; however, little room exists for single-ended sensing in the modern paradigm of performance-driven DRAM design.

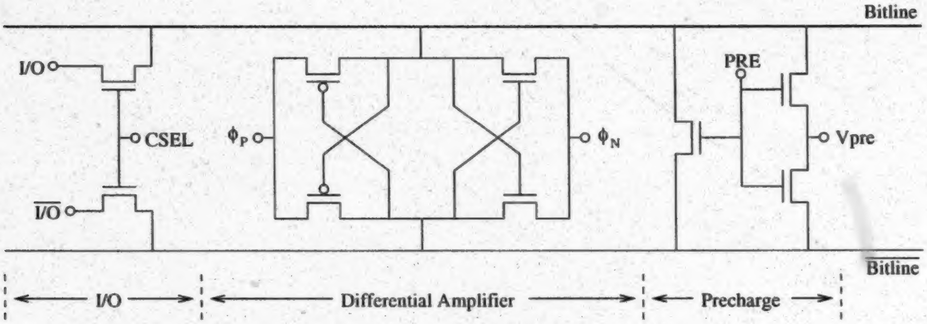


Figure 2.3: Positive Feedback Differential Sense Amplifier [16]

2.3.4.2 Differential Voltage Sensing

The most popular sensing technique in modern DRAMs is differential voltage sensing. The advantages of using differential sense amplifiers are that they are non-inverting, they reject common mode noise, they are insensitive to process variations, they are very sensitive to small signals, and they can operate very quickly. Furthermore, by using positive feedback in a differential amplifier, reading and restoring is merged into a single operation, and performance is improved [10, 39].

Figure 2.3 shows a positive feedback differential sense amplifier, complete with precharge, equalization, and I/O circuits. Before a read operation is performed, the bitlines are precharged (normally to $V_{DD}/2$) and equalized by the precharge and equalization circuit. Once a signal is developed on a bitline from a memory cell, the feedback differential amplifier is activated by asserting ϕ_P and $\overline{\phi_N}$. The bitlines are driven to full complementary V_{DD} and ground levels based on the signal from the open memory cell, and in the process the cell itself is restored to its original voltage level. As the sense operation completes, the I/O transistors are activated and the read voltage is transferred to a data bus.

2.3.4.3 Direct Sensing

Direct sensing is a technique that separates the read (output) lines from the write (input) lines. It actually uses a conventional differential voltage sense amplifier to detect the stored cell voltage; however, it decouples the I/O lines from the bit-

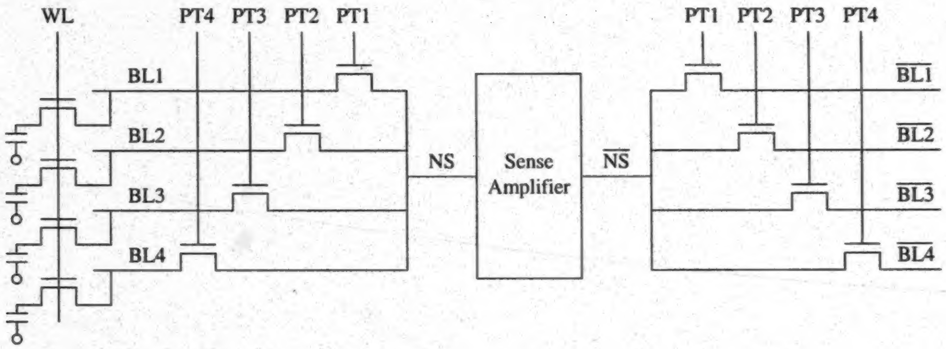


Figure 2.4: Time Multiplexed Sensing Structure (adopted from [12])

lines, allowing significantly faster sensing at the expense of additional area. When a sense operation occurs in a direct sensing scheme, two transistors whose gates are connected to the bitlines generate a small differential current on the I/O lines. This current is detected immediately and amplified, even before the bitlines complete their amplification process. The result is a very high speed sensing and I/O operation [14].

2.3.4.4 Time Multiplexed Sensing

Time multiplexed sensing is a technique that allows performance to be sacrificed for a substantial gain in density and some improvement in noise rejection.

In time multiplexed sensing, a single sense amplifier services multiple bitline pairs. Each pair is sensed (and restored) in sequence, and the result of each sense operation is transfered out of the memory core via a global bitline or data bus. This sharing of a single sense amplifier between multiple bitlines significantly reduces the sense amplifier area in a DRAM, which in turn reduces the area of the DRAM chip as a whole [40, 12].

Figure 2.4 shows the basic bitline and sense amplifier organization for a time multiplexed sensing scheme. In a read and restore operation, the wordline WL is activated, and then the bitline access transistors are activated one pair at a time to allow sensing of each individual bitline pair at a time.

Time multiplexed sensing is a very useful technique in the design of semiconductor file memories, as is discussed in section 2.4.2.

2.3.5 Noise in DRAMs

Noise management is a crucial part of DRAM design. Excessive noise will prevent a DRAM from functioning correctly, rendering it completely unusable. In this dissertation, the term "noise" refers to everything that degrades signal integrity, even if these noise sources are deterministic.

Noise within the DRAM array is generated predominantly from three sources: capacitive coupling to the bitlines from neighbouring conductors, process variations between bitlines (including sense amplifier variations), and the intrinsic offset of sense amplifiers [14]. For a DRAM to function properly, the noise voltage must be less than the worst case noise margin within the array.

The noise margin can be calculated as follows:

$$v_{nm} = v_{signal} - v_{leakage} - v_{critical} \ , \tag{2.2}$$

where v_{signal} is the maximum signal voltage available on a bitline, $v_{leakage}$ is the voltage reduction due to cell leakage between refresh cycles, and $v_{critical}$ is a voltage corresponding to the soft-error critical charge for a cell. This equation can be also be written in terms of explicit design variables,

$$v_{nm} = \frac{C_s}{C_b + C_s}\left[\frac{V_{DD}}{N} - \frac{I_l \Delta t_{r_{max}}}{C_s} - \frac{Q_c}{C_s}\right] , \tag{2.3}$$

where C_s is the capacitance of a cell, C_b is the capacitance of a bitline, N is the number of stored voltage levels (2 for standard DRAM), I_l is the cell leakage current, $\Delta t_{r_{max}}$ is the maximum time between refresh cycles, and Q_c is the soft-error critical charge for a cell.

Then for the DRAM to function correctly, the condition

$$v_{nm} > v_{noise} \tag{2.4}$$

must be satisfied, where v_{noise} is the total noise voltage due to the factors described above as well as thermal noise.

2.3.6 Leakage in Sub-0.13-μm DRAMs

Leakage is becoming a very important consideration as DRAM devices scale to increasingly small sizes, due to the effects of leakage on power consumption and noise margins. As DRAMs move below the 0.13-μm mark, the leakage sources that currently affect DRAM designs become greater, and new leakage sources come into play that were not significant before.

There are four significant leakage mechanisms in sub-0.13-μm DRAMs:

1. **Weak inversion (subthreshold) current**. The primary method of leakage in today's transistors.

2. **PN junction reverse bias current**. Responsible for leakage from the source or drain of a cell transistor to bulk. Composed of diffusion, electron/hole pair generation, and band-to-band tunneling (BTBT) [23], this form of leakage is highly temperature dependent.

3. **Gate-induced drain leakage (GIDL)**. A current from the drain to the substrate that occurs with negative wordline bias, aggravated by carrier traps in the oxide. The consequence of this form of leakage is that negative wordline biasing is only effective when effort is made to eliminate traps in the gate/drain overlap region of a transistor during DRAM manufacturing [4].

4. **Gate dielectric oxide tunneling**. Oxide tunneling in cell transistor gates as well as in cell capacitors is significant for $t_{ox} < 4$ nm, which is the case for most processes from 0.13-μm and smaller. Oxide tunneling is composed of direct tunneling and Fowler-Nordheim tunneling, the latter being insignificant under normal operating conditions. Direct tunneling can be further decomposed into conduction band and valence band tunneling [20], and for transistors, into gate-to-channel and edge-direct tunneling [5].

A number of techniques have been developed to combat leakage and its effects. Modern DRAMs use high threshold cell access transistors in the memory array to reduce subthreshold leakage during inactivity. The practise of leaving bitlines precharged to $V_{DD}/2$ during inactivity also reduces subthreshold leakage. A third technique to reduce subthreshold leakage is to apply a negative wordline bias during inactivity; however, this practise can cause gate-induced drain leakage, as mentioned above.

PN junction reverse bias current can be controlled by reducing the drain-bulk and source-bulk junction temperatures, which can be achieved with low-power circuits and packaging with low thermal resistance [14]. Beyond that, PN junction current can only be reduced through careful control of the geometry, doping concentration, and defect density of the source or drain regions of critical transistors.

Until recently, oxide tunneling has been insignificant in DRAMs, and designers have ignored it. However, as oxide thickness is reduced below 4 nm, tunneling is becoming an important source of leakage [31]. In fact, it has been stated [14] that dual-gate-oxide-thickness DRAM processes will soon be required, with a thin oxide in peripheral transistors to achieve high performance, and a thicker oxide in the memory array to reduce leakage and improve reliability.

For further discussion of the leakage mechanisms that can affect sub-0.13-μm semiconductor devices, the reader is referred to Roy and Prasad's book on low-power CMOS [32].

2.3.7 DRAM Manufacturing Process Considerations

DRAM manufacturing processes are different from typical CMOS processes in a number of special ways. The difference between the two is motivated by the need for DRAM chips to be denser and more reliable than they would be if a logic CMOS process were used. However, the use of a special process means an added responsibility for designers: it is necessary to consider the effects of these differences when designing DRAM circuits.

DRAM arrays are fabricated using NMOS-only technology with a different layer set from the rest of the chip. There is no space for N-well regions in the tightly packed cell array[3]. Finer than normal line pitches are possible for wordlines and bitlines due to the regular structure of the array. At the same time, there is usually a limited number of upper level metal layers available, and those that are available have a larger pitch than the bitlines or wordlines. Additional layers added to DRAM processes include an extra polysilicon layer to allow capacitor nodes and/or different wordlines to coexist in the same region together, and polycide and tungsten layers as bitlines or as the common cell plate in stacked capacitor cells [38].

The most interesting aspect of the DRAM array fabrication process is the creation of cell capacitors. Process engineers have developed many different cell structures, but all of the structures that have been used in commercial DRAMs are based on either a planar, trench, or stacked capacitor configuration. Planar capacitors are simple but inefficient in terms of area. They use two parallel layers as electrodes with an SiO_2 dielectric. Today these have been replaced by the trench and stacked configurations. Trench capacitors normally have a polysilicon electrode buried in the substrate, with a dielectric between the two. Stacked capacitors normally have a polysilicon electrode deposited vertically over top of the wordlines, with an SiO_2 or Ta_2O_5 dielectric. Expanding in the vertical direction greatly increases the capacitance of the capacitors by increasing their parallel area.

The special nature of DRAM processes has several consequences on chip design. Most importantly, circuits must be designed to minimize the effects of process variations. The density and complexity of the memory array contribute to small differences between each cell. If care is not taken when designing the array and its supporting circuitry, these process variations can translate into a significant reduction in noise margins. Another consequence, related to the large scale repetition of the cell pattern in an array, is that dummy rows and columns are needed at the edges of each sub-array to avoid photolithography problems that can occur at these edges [16]. Finally, the fine pitch of lower metal and polysilicon layers, combined with the limited pitch of upper metal layers, means that the designer must carefully plan which signals need to be routed through the array, and design sense amplifiers that satisfy the array's strict design rules.

---

[3] An exception to this is in trench capacitor structures that have an N-well region buried under the entire array, which is used as a common cell plate.

2.3.8 Cost and Economics

Economics are an important part of the DRAM industry and DRAM design. DRAM has been mass-produced to the extent that its price, like that of automobiles and cattle, is controlled more by international economic conditions than by DRAM manufacturers. Cost control and reduction in DRAM is therefore very important, to the extent that designers must consider the cost repercussions of their decisions in order to develop competitive products.

It is well known that the popularity of DRAM is founded in its low cost per bit relative to other semiconductor memories. However, if one assumes that density is maximized in all cases, there are other factors that influence the cost of DRAM chips.

1. **Design Effort**. Greater design effort increases the cost of DRAM chips. As is evidenced by the increasing complexity of commercial DRAM designs, the increased non-recurring cost of design effort is insignificant when prorated over production quantities of chips [6].

2. **Chip Size**. Chip size is influenced by cell density, but also by the area of peripheral circuits. Designers need to keep the area of the peripheral circuitry to a minimum.

3. **Processing Complexity**. Increasing the number of processing steps that are needed, as well as the complexity of each of those steps, reduces production throughput. This reduced throughput translates into an increased production cost [43]. Designers therefore need to consider the cost tradeoffs when adding additional layers or other process steps to a DRAM architecture.

4. **Production Yield**. Yield is a very important cost factor, because the designer has some control over it. Yield is influenced not only by the quality

of manufacturing process used, but also by other factors such as memory array organization and the extent to which redundancy is applied. Determining an appropriate tradeoff between performance, density, and yield is a complex problem that depends heavily on process- and design-specific variables.

5. **Testing**. Long test times reduce chip throughput, increasing cost. The cost in increased die area to implement on-chip testing circuitry is generally prohibitive, so the designer has little control over the cost of testing beyond providing an adequate data rate.

6. **Wafer Size and Packaging**. Wafer size and packaging are two additional factors that affect DRAM cost, factors over which the designer has little control. The only way a designer can influence the cost of packaging is through power reduction in the memory. Power reduction is usually motivated more by the need to improve reliability and to reduce energy usage in portable devices than by cost considerations, however.

2.4 File Memory

File memory refers to the use of inexpensive, high-capacity memory for block storage and access of data, in applications where random access may be convenient but is unnecessary. File memory is suited for applications in which the performance of current memory technology is greater than that demanded by the application, including mobile computing systems and portable data storage [48]. In this thesis we are primarily interested in the use of file memory as a cache level in desktop and server computer systems. This section describes how file memory can be used in a computer memory hierarchy and presents some examples of file memory implementations.

2.4.1 File Memory in a Memory Hierarchy

A computer memory hierarchy is composed of multiple levels of memory. Each level in the hierarchy has a greater latency, lower cost per bit, and consequently a larger capacity than the level above it. The use of memory hierarchy allows a computer system to enjoy the advantages of a small amount of expensive, high performance memory while having the large capacity offered by inexpensive, low performance memory [13].

Figure 2.5 shows a typical computer memory hierarchy. The fastest memory in a computer system is in the register file, which is usually an SRAM-based memory. Because this memory uses precious area on the same die as the microprocessor, it is very expensive. At the other end of the hierarchy is disk, which is normally implemented as a magnetic hard drive. Disk drives are inherently slow due to the mechanical nature of their operation, but have a very low relative cost per bit. Between these extremes, various storage levels have been introduced over the past decades in order to improve system performance and control cost.

In the typical computer memory hierarchy shown figure 2.5, there is a substantial gap between main memory and disk in terms of latency and cost. Recent research by Koob has demonstrated that the use of file memory as "Extended Storage Disk Cache" (ESDC) to fill this gap in the hierarchy can improve the performance of a computer system at no increase in cost[4]. Using one benchmark, it was found that a 36% increase in read throughput and a 35% improvement in write operation time could be achieved at no additional cost in the system [18].

Based on Koob's results, there is strong motivation to include an extended storage level into computer memory hierarchies. Unfortunately, unlike the levels surrounding the memory hierarchy gap, it is difficult to develop a suitable technology with which to fill the gap.

2.4.2 Prior File Memory Implementations

A few file memory implementations have been developed that are aimed at making the use of an extended storage scheme economically feasible. The implementations described here are all DRAM-based, but apply different technologies and innovations to create file memory.

---

[4] Equivalently, the cost of a computer system can be reduced with no performance penalty.

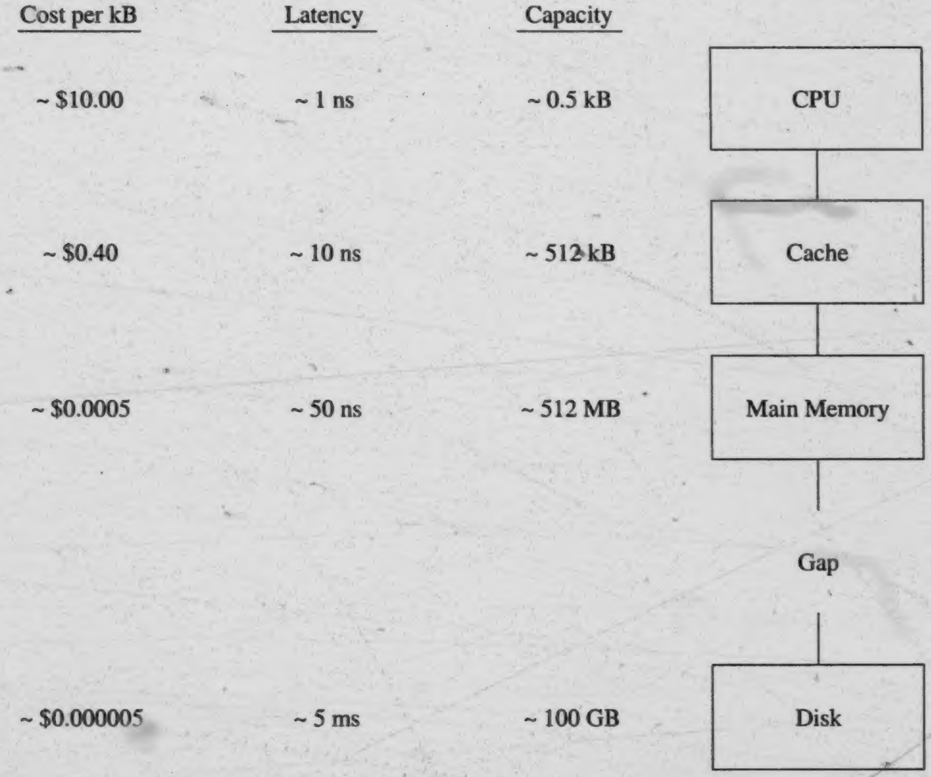

Cost per kB
Latency
Capacity
~ $10.00
~ 1 ns
~ 0.5 kB
CPU
~ $0.40
~ 10 ns
~ 512 kB
Cache
~ $0.0005
~ 50 ns
~ 512 MB
Main Memory
Gap
~ $0.000005
~ 5 ms
~ 100 GB
Disk

Figure 2.5: Common Computer Memory Hierarchy with Approximate Cost, Latency, and Capacity (values taken from Hennessy and Patterson [13] and the ITRS [36])

2.4.2.1 MLDRAM

Various multilevel DRAM (MLDRAM) chips have been developed over the last 15 years [1] in an attempt to improve the density of DRAMs without the use of new process technology. The concept behind MLDRAM is the storage of multiple analog voltage levels in a DRAM cell, rather than the usual two digital voltage levels of ‘0’ and V_{DD}. If N different voltage levels are stored in each DRAM cell, then the memory can hold $log_2 N$ bits per cell. Ideally, this is also equivalent to a density increase of $log_2 N$; however, additional circuitry is normally required to support the use of multiple voltage levels, reducing the actual density increase.

The goal of most MLDRAM designers has been to compete with DRAM in terms of performance, power consumption, and robustness using significantly less area per bit. Varying degrees of success have been achieved, but the performance of MLDRAM designs has not been sufficiently competitive with DRAM to justify industry adoption. This has led some researchers to propose the use of MLDRAM for file memory [25, 2].

MLDRAM is well suited for use as file memory because, even though it can't offer the performance of DRAM, it can be much more economical. It is possible that MLDRAM will have a commercial introduction for use specifically as file memory in the coming years.

2.4.2.2 Modified DRAM

Another approach to file memory design is to modify the architecture of regular DRAM by making tradeoffs that increase density. Such tradeoffs might include reduced performance, increased complexity, higher power consumption, fewer access modes, or reduced robustness. Because the ultimate goal of file memory is to be more economical than DRAM, density can also be traded off in the interest of reduced cost.

A notable design that uses this modified DRAM approach was developed by Sugibayashi et al. with NEC Corporation [40]. The design is a 1-Gb DRAM that uses a time multiplexed sensing scheme, trading off performance and random access for an increase in density. As described earlier, time multiplexed sensing allows the use of fewer and smaller sense amplifiers than DRAM, which in turn allows a smaller overall chip size. The design also reduces the production cost of DRAM by using a "Flexible Multi-Macro" architecture. In this architecture, a chip is formed from four identical macro blocks. When a wafer is fabricated with many repeated macro blocks, the saw lines between groups of four blocks can be adjusted to maximize yield. The penalty for this flexibility is a 2% area overhead, but the result is a 7% increase in yield, assuming a mature manufacturing process that can achieve 90% yield on each macro block.

Another file memory design that preceded Sugibayashi's design uses NAND-structured cells to increase the density of the memory array. This design, developed

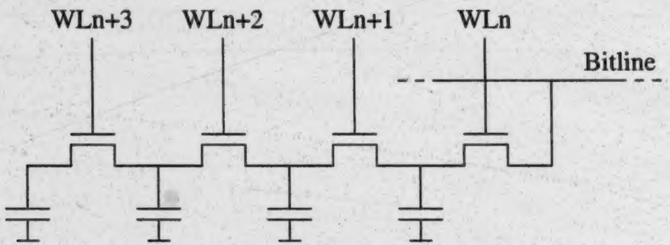


Figure 2.6: NAND Structured Memory Cells [12]

by Hasegawa et al. with Toshiba Corporation [12], enables the use of very small cells. Because the cells are organized into groups of four in series, the average bitline contact and isolation area per cell is reduced to 1/4 of that in a standard DRAM. The NAND structure of these cells, shown in figure 2.6, is responsible for the cell grouping capability.

The NAND-structured cell array combined with a time multiplexed sensing scheme necessitates two dimensions of sequential sensing. The sensing of bitlines is performed using time multiplexed sensing as described previously. However, each bitline is sensed once for each cell in a NAND structure. This sensing operation greatly hinders performance, but the increase in density that it allows makes the DRAM quite economical for use as file memory.

2.4.2.3 ECC and Bad Block Marking

A unique technology has recently been investigated by Wickman of the University of Alberta. Wickman proposes a technique in which "the requirements of modern memory are relaxed in order to increase the equivalent yield and decrease the average cost per working bit" [48], through the application of error-correcting codes (ECC) and bad block marking (BBM). While the idea of using ECC and BBM with semiconductor memory is not new, their use in file memory is novel.

Commercial DRAM chips must be 100% functional for them to be sold in the computer market. This means that many manufactured chips that have only a few non-functional and non-replaceable bits are discarded and have no value. A system similar to ECC in digital communications or BBM in hard disk drives can be applied to DRAMs to improve their yield, lowering their cost. These DRAMs can make good file memories, having only the disadvantages of reduced performance and a loss of random-access capability.

Wickman found that tens of thousands of defects per chip can be tolerated if ECC is used in combination with redundancy, or if BBM is used, without an increase in cost per bit compared with traditional redundancy [49]. As a result, more devices per wafer can be salvaged by using ECC or BBM.

A favourable aspect of ECC and BBM is that they could be applied to the file memory implementations described in the previous sections to lower their cost. The area overhead of ECC and BBM is quite small, which makes combining them with other technologies all the more promising.

2.5 Summary

DRAM remains the most widely embraced semiconductor memory product in the world today, due to its low cost per bit and its superior performance. With the existing industrial and research infrastructure that is in place to improve and proliferate DRAM, it is unlikely that its worldwide production and consumption will soon subside.

As no reasonable competing technologies have emerged in contention for use as file memory, it is logical to consider DRAM-based solutions for filling the memory hierarchy gap. While the file memory implementations presented in this chapter are successful in reducing the cost per bit of DRAM by introducing innovations and tradeoffs, the full extent to which DRAM density can be increased has not yet been reached.

Chapter 3

Proposed Dynamic File Memory Architecture

3.1 Overview

This chapter proposes a new architecture for creating economical DRAM-based file memory. The architecture is based on the use of very small, simple sense amplifiers, as well as an open bitline array organization. A number of novel techniques and tradeoffs are presented for coping with reduced-complexity sense amplifiers, as well as for reducing the production cost of the memory. Such techniques include reducing bitline length, adding separate read and write select lines, using a two-pass write operation to reduce coupling noise, allowing for serial I/O, and employing a unique reference scheme that we call the "Wine Cellar Technique."

Important aspects of the architecture are described in detail. For other parts, where the specific implementation is not important, only a general description is given of how that part fits in to the grand scheme. Ultimately, the focus of this chapter is on how a minimal sense amplifier can be designed and integrated into a feasible file memory architecture.

3.2 Introduction to the Proposed Architecture

3.2.1 Concept

The number of sense amplifiers per chip has increased with each DRAM generation. While the number of cells per sense amplifier must remain nearly constant[1] so as not to increase bitline capacitance, the total number of cells increases exponentially with each generation. Large scale DRAMs, such as one proposed 4-Gb DRAM [50], require several million sense amplifiers. In existing and past commercial designs, the memory array generally only occupies about 55-65% of chip area due to the overhead of sense amplifiers and other peripheral circuitry [14].

The fundamental idea behind the proposed architecture is the use of very small sense amplifiers, amplifiers that use only a single transistor to read data from a bitline. The use of very small sense amplifiers can lead to a memory design that has a significantly smaller area than conventional DRAM. There are two reasons for this. The first is that the sense amplifiers in a DRAM typically occupy between 10-15% of the overall area of the chip [16]. Therefore a reduction in the area of each sense amplifier results in a reduction in overall chip size. The second reason is that small sense amplifiers allow the use of the noisier but denser open bitline array organization. If sense amplifiers are smaller, there can be more of them in the array core. With more sense amplifiers in the array core, the number of cells per bitline can be reduced, thereby improving noise margins within the array. These increased noise margins counteract the increased noise of an open bitline array.

A disadvantage of using a single transistor sense amplifier is that performance is greatly degraded. As explained in section 2.3.4, a major advantage of the conventional DRAM sense amplifier is that its positive feedback configuration makes it operate very fast. A single transistor amplifier, whose operation is explained in detail later in this chapter, has no feedback, and must drive a highly capacitive data bus. However, this tradeoff of performance for density is exactly the tradeoff needed for a file memory design to fill the gap between main memory and disk in the computer memory hierarchy. Another disadvantage of the single transistor sense amplifier compared to conventional DRAM sense amplifiers is that cell restoration is more complicated. Conventional DRAM sense amplifiers can refresh cells simply by being activated, but single transistor amplifiers can not do the same. This problem, and how it is dealt with, are discussed in more detail later in this chapter.

---

[1] Commercial DRAMs generally used 512 cells per sense amplifier from the 1-Mb generation through the 256-Mb generation, and 1024 cells per sense amplifier since then [8, 14].

A unique advantage of using a single transistor sense amplifier is that a memory can be made to support the storage of multiple values per cell with relatively little overhead. Conventional sense amplifiers, such as those on which existing multilevel DRAM designs are based, force the bitlines to full supply voltage levels during sensing. This means complicated structures are required to ensure their compatibility with values other than '1' and '0' [1]. However, a single transistor sense amplifier maintains the exact value of analog data much farther along the memory datapath. Therefore, assuming that noise margins remain satisfied, relatively little additional circuitry is required to create a multilevel file memory based on a single transistor sense amplifier memory architecture.

3.2.2 Requirements

There are several important basic requirements for the design of a DRAM-based semiconductor file memory that uses a minimal single transistor sense amplifier. These requirements stem from the need for the memory to function correctly and the need for it to fit appropriately into the memory hierarchy gap shown in figure 2.5. They are outlined below.

Correct Functionality and Reliability An obvious requirement in any modern computer subsystem is functional correctness. The file memory must be expected to operate correctly, and additionally it must be reliable under the same operating conditions as DRAM.

High Density The primary requirement for a semiconductor file memory is that it must be denser than existing semiconductor memories. This is necessary so that the file memory fits appropriately into the memory hierarchy gap between main memory and magnetic disk, having a higher capacity and lower cost per bit than any other semiconductor memory.

Adequate Performance The memory must have faster access and block transfer times than magnetic disk for it to satisfy the speed requirements imposed by the computer memory hierarchy. However, the memory can operate much slower than DRAM and still fill the memory hierarchy gap very successfully.

Inexpensive Manufacturing The memory must be able to be manufactured easily, and for approximately the same cost as for DRAM. This is required so that the cost benefits of increased density are not lost due to additional manufacturing complexity. This requirement implies the need for it to be possible to fabricate the memory in an existing DRAM process, or in one with minimal modification, using very few extra masks or special process techniques.

Adequate Yield Another requirement resulting from the need to minimize the cost of the file memory is the need for high yield. DRAM yield in volume production is generally around 85%, depending on the maturity of the process [36]. The file memory needs to attain a similar yield if the cost savings from increased density are to be fully exploited. A consequence of this requirement is that circuits in the memory must be designed so as to be insensitive to process variations.

Reasonable Power Consumption As with all modern semiconductor devices, it is desirable to have reasonable power consumption. This is particularly true in memory devices, which can suffer from reliability problems from large amounts of heat dissipation. Although reducing power consumption is not a primary goal in this work, it is nonetheless considered and analyzed where appropriate.

In the work presented in this chapter, one additional goal is considered throughout the development of the architecture. This goal is for the architecture to support multiple-level storage with minimal changes. This way, the architecture remains as general as possible, and implementation technology becomes the only real constraint on potential density improvement.

3.2.3 Basic Operation

Before presenting the detailed characteristics of the various components of the proposed architecture, it is constructive to describe the basic operation of the entire memory. The memory functions similarly to conventional DRAM in most ways. From an abstract perspective, it accepts addresses as input and provides data as output. However, the memory uses sequential block access, rather than random access, and there are fewer addressable blocks than there are storage cells in the memory.

Within the memory array are typical dynamic storage cells that operate identically to the cells in DRAM. When a sub-row of data, which constitutes a single block unit of sequential data, is to be read from a sub-array, the sub-wordline corresponding to that sub-row is activated. All of the cells in that sub-row transfer their data to their respective columns. Unlike DRAM, however, the column data is not detected, amplified, and latched by digital sense amplifiers. Rather, the column data remains dynamically stored on the bitlines as an analog voltage. Each column is then read and amplified in sequence with dedicated sense amplifiers that feature single transistor read amplifiers.

Each sense amplifier along a single row is connected to a common data bus, as depicted in figure 3.1. This bus carries an amplified analog representation of the column data (in the form of a current) from the array to the periphery, one column at a time. In the peripheral circuitry for each bank is a data converter that converts the analog signal into a digital representation of the original stored data.

In order for a data converter to interpret the analog signal data, reference values are required. These references are read from additional columns in the active sub-row, in the same way as the rest of the data. The references are read and sampled before any other columns are read. They are sampled and compared to the analog read value from each column to determine which value was stored. The read data is then transferred to an SRAM cache that stores it for restore and I/O purposes.

Once the entire sub-row has been read, the data is rewritten to the sub-row sequentially while the sub-row remains active. After the entire sub-row has been restored, including the stored reference values, it is deactivated. Writing to the

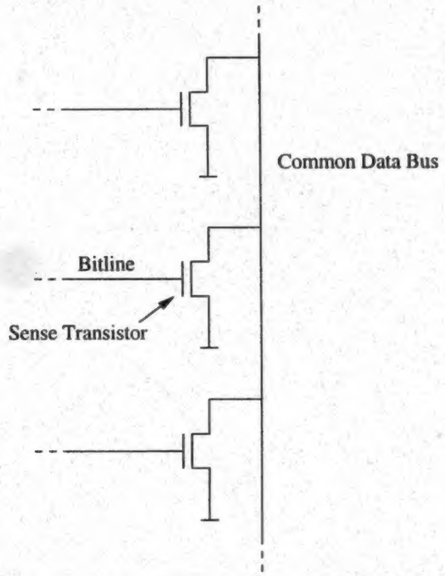


Figure 3.1: Conceptual Diagram of Sense Amplifier Transistors Along a Data Bus

memory occurs in a similar fashion, except the data that populates the SRAM comes from external I/O rather than from the array.

The operation of the proposed architecture is described in much greater detail in section 3.3.

3.3 Description of the Proposed Architecture

The proposed dynamic file memory architecture consists of numerous sub-arrays of 1T DRAM cells connected to regular control and I/O circuitry. In this regard, it exhibits a similar organization to conventional DRAM. However, the sense and restore circuitry as well as the I/O circuitry are substantially different. The proposed architecture is modular, and each component of the architecture contributes in some way to satisfying the requirements of section 3.2.2. At the same time, there is flexibility in each part of the overall scheme that allows a designer the opportunity to make tradeoffs in a final implementation.

3.3.1 Array Architecture

An open bitline array organization is the best choice for a DRAM-based file memory that uses single-ended sensing. There are three major reasons for this. First, the open bitline organization is the most dense possible organization for a cell array. Second, the proposed architecture has numerous characteristics that make the noise immunity of a folded array either unnecessary or not useful. These characteristics include reduced voltage swing on read, time multiplexed sensing, the ability to have shorter than typical bitlines, and the use of single-ended sense amplifiers. Finally, recent research indicates that as DRAMs scale to smaller sizes, the advantages of a folded bitline scheme are reduced. These reasons are explained in detail below.

As described in section 2.3.3.1, the open bitline organization allows $6F^2$ memory cells, as opposed to the $8F^2$ cells of the folded bitline organization. This represents a 25% reduction in density just in the memory cells by using open bitline. Moreover, a further reduction is achieved with open bitline because bitline twisting is not useful. The area that is used for twisting in a folded memory core is freed up, increasing the area savings for open bitline.

Noise generation is limited in the proposed architecture. One way this is accomplished is by avoiding a full voltage swing during read operations. As will be described in section 3.3.2, restore operations in the proposed sensing scheme do not occur until a full sub-row has been read. This prevents some noise from being coupled between bitlines during a read, reducing the need for the bitline twisting allowed by a folded bitline organization.

File memories that use a time multiplexed sensing scheme are particularly immune to the array noise that plagues open bitline arrays in conventional DRAMs. As stated by Sugibayashi et al., “The array noise impact is not critical for file memories because the additional delay required in waiting for the noise signal to decay affects only the first access time and the precharging time.” [40]

The use of a single-ended sense amplifier, which is a key characteristic of the proposed architecture, negates any advantage that remains in using folded bitlines, since the amplifier only has one input. However, the use of a very small sense amplifier makes it possible to subdivide each bitline more than in a typical DRAM, while still improving the overall density of the memory. By subdividing the bitlines more than usual, the parasitic bitline capacitance is reduced, along with the coupling capacitance between neighbouring bitlines. This increases noise margins despite the use of an open bitline organization.

The random array noise that is canceled by a folded bitline array is becoming less important as arrays are made smaller through the use of newer processes. In the proposed architecture, as with traditional open bitline arrays, a reference bitline is still needed for sensing. Although that reference bitline is not immediately adjacent to the active bitline, as it would be in a folded array, it remains physically not far away. As arrays scale to smaller sizes, active and reference bitlines come physically closer together, which reduces the noise imbalance between the two [41]. Noise from inter-bitline coupling, which is canceled by bitline twisting in a folded array, is also expected to diminish as arrays are miniaturized [29]. This result suggests that the need for the folded bitline structure will diminish as well.

In general, the techniques proposed in this chapter will support any array architecture, possibly with minor modifications. However, with density as a primary goal, and based on the arguments presented above, an open bitline structure is the best choice for use in the proposed architecture.

3.3.2 Sensing Scheme

The most differentiating aspect of the proposed dynamic file memory architecture is the sensing scheme, which uses a single transistor to sense data on a bitline. Such an amplifier provides the greatest density improvement and flexibility in array design because it leads to a very compact layout and can exactly sense any voltage that appears on a bitline. It also presents a number of design challenges, in terms of noise management and core organization, that this chapter attempts to address.

3.3.2.1 Sense Amplifiers

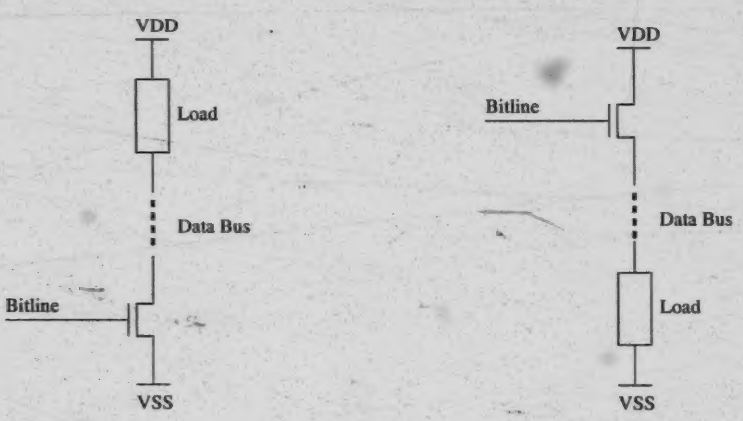
The characteristics that make a good single transistor read amplifier are high gain, high input impedance, large valid input swing, small layout area, and good noise immunity. To attain these characteristics, a single transistor amplifier can be configured as a common source or common drain transconductance amplifier, with the transistor gate connected to a bitline as the input to the amplifier. A common gate configuration is inappropriate due to its low input impedance [28]. Either an NMOS or PMOS transistor can be used as the sense transistor. This leads to four distinct configurations that can be considered, as shown in figure 3.2.

A common source transconductance amplifier (CSTA) with an NMOS transistor (figure 3.2(a)) provides high gain and allows an input swing from V_{th} to V_{DD}. The NMOS source is grounded and the current through the transistor is read on the drain side. The drain can be loaded with a passive or active load, or can have zero (or near zero) impedance. Some loading is unavoidable due to a resistive data bus and the presence of a global sense amplifier on the bus.

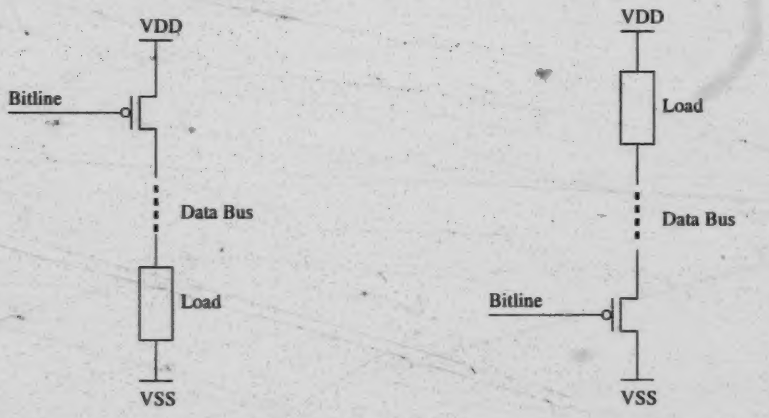
In the common source configuration, the body effect is a factor if a degenerative load (such as a column select transistor, as is discussed later) is used at the source of the sense transistor. The body effect will cause the threshold voltage of the sense transistor to increase, effectively reducing the valid input swing. This swing reduction is relatively small, however, and consistent throughout the entire memory array. A relatively wide output swing can be achieved with proper biasing.

A common drain transconductance amplifier (CDTA) with an NMOS transistor (figure 3.2(b)) is also a valid configuration, with similar characteristics to those of the CSTA. The NMOS drain is connected to V_{DD} and the current through the transistor is read on the source side. Like the CSTA, the source can be and effectively must be loaded, which introduces the body effect.

There are two significant disadvantages to an NMOS CDTA in comparison with an NMOS CSTA for use as a sense amplifier. The first is that if a column select transistor is used at the drain of the sense transistor, there will be a threshold drop from V_{DD} to the drain of the sense transistor. This will cause a significant reduction


VDD
Load
Data Bus
Bitline
VSS
VDD
Bitline
Data Bus
Load
VSS

(a) NMOS CSTA (b) NMOS CDTA


VDD
Bitline
Data Bus
Load
VSS
VDD
Load
Data Bus
Bitline
VSS

(c) PMOS CSTA (d) PMOS CDTA

Figure 3.2: Four Potential Read Amplifier Configurations

in the output range of the CDTA. The second disadvantage is that source loading, and therefore the body effect, is less predictable. This is very undesirable in the proposed architecture, where matching the characteristics of each sense transistor is of great importance.

As an alternative to using an NMOS sense transistor, a PMOS sense transistor could also be used in either a CSTA or CDTA configuration (figures 3.2(c) and 3.2(d)). However, there are a number of disadvantages to this approach. The most obvious disadvantage is the increased area required by a PMOS transistor. Substan-

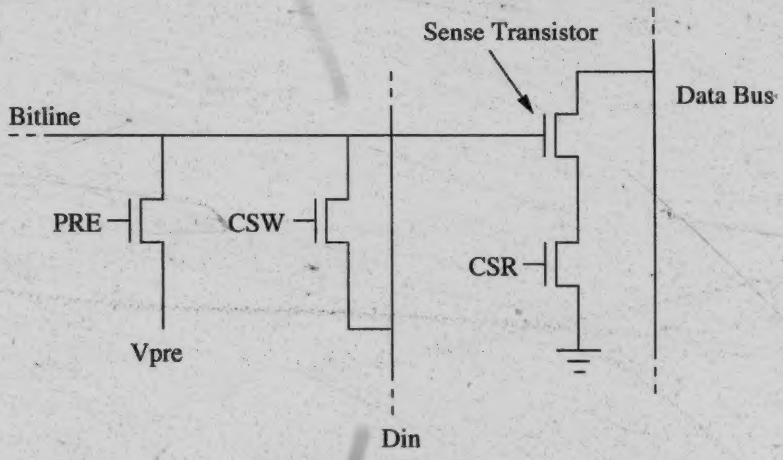


Figure 3.3: Four-Transistor Sense Amplifier Configuration

tial area is made necessary by the addition of an N-well region and the need for a larger PMOS device to counteract the lower carrier mobility of P-type material. If PMOS devices are avoided altogether in the memory core, then no N-well region is necessary, and NMOS transistors can be fabricated in the same way as cell access transistors. This can result in significant area savings. Another disadvantage is that DRAM processes normally maintain much tighter control over NMOS transistor characteristics than over those of PMOS transistors [10]. This means that PMOS sense transistors would exhibit more variation from one another than would NMOS sense transistors. These disadvantages must be weighed against any potential advantage of the lower-biased valid input range (0 to (V_{DD} - V_{th})) that the use of PMOS enables.

Support circuitry must be added to the single transistor amplifier to allow fully-functional read, write and restore operations. This circuitry must add as little area overhead as possible. The proposed circuitry, complete with amplifier, is shown in figure 3.3. The figure shows an NMOS CSTA with a column select read (CSR) transistor at the amplifier source. The other transistors are for write and precharge operations, and their specific purposes are described in the next subsections.

This sense amplifier requires very little silicon area to implement, as it only has four transistors and they are all NMOS transistors. The transistor sizing must be

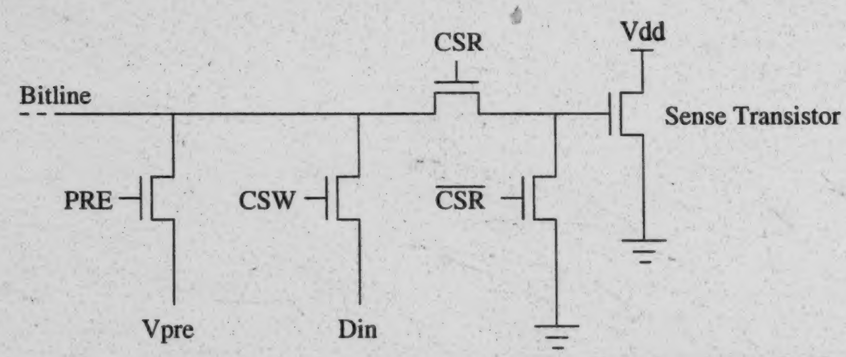


Figure 3.4: Alternate Sense Amplifier Configuration

chosen to minimize the size of the sense amplifier while ensuring reliability and correct operation. For a file memory design there is no advantage to increasing transistor size only to improve performance, so the write and precharge transistors can be minimum sized. The sizing of the read and CSR transistors is more important. The width of these transistors should be large enough that the read amplifier has a high enough gain to satisfy signal margin requirements. The length of these transistors should be large enough that small variations in length have negligible effect. A common practice in DRAM sense amplifiers is to use a length of 1.2 to 2 times the minimum allowable length [8].

An alternative to the four-transistor design is shown in figure 3.4. This design eliminates the need for a CSR transistor in series with the sense transistor, but introduces a series transistor on the bitline and requires a fifth transistor to pull the read amplifier low during inactivity. This configuration requires a boosted read select signal to provide a full range of input to the read transistor. Furthermore, variations in the series read select transistor add to the uncertainty in sensed data. These effects, combined with the area penalty of using an extra transistor, make this sense amplifier undesirable.

Based on the discussion above, it is apparent that a four-transistor NMOS CSTA, hereafter referred to as the 4T sense amplifier, offers a good set of characteristics for our file memory implementation. Therefore, this amplifier was chosen for use in the proposed architecture. The proceeding sections describe how it is used for

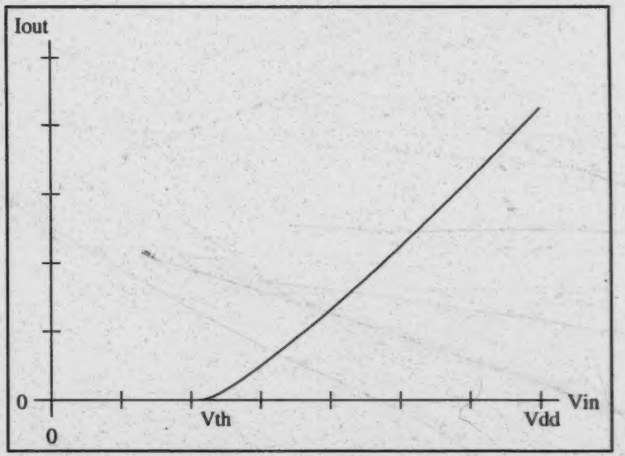


Figure 3.5: Approximate I/O Characteristic for the NMOS 4T Sense Amplifier

read, write, and restore operations.

3.3.2.2 Read Operation

A read operation begins by precharging the bitline, which is done by activating the PRE control line seen in figure 3.3. The precharge voltage is set to a carefully controlled amplifier bias voltage, which does not need be $V_{DD}/2$. The purpose of this bias is to place the amplifier appropriately in its useful operating range. In the case of the 4T sense amplifier, the amplifier can not distinguish bitline voltages around and below V_{th} because there is little to no output distinction between different input levels, as shown in figure 3.5. Therefore, the bitline should be biased such that its voltage remains above V_{th} after charge sharing has occurred. There is in fact an optimal bias point that allows the maximum possible amplifier output swing for a given cell ratio. Interestingly, this bias point is not necessarily in the exact center of the amplifier's useful input range. A complete analysis of the optimal bias point problem is presented in section 4.6.

With the bitline precharged, a sub-wordline is activated and the stored voltage in the selected memory cell is mixed with the bitline bias voltage through capacitive charge sharing, as governed by equation 2.1. The resulting voltage is maintained on the floating bitline while all bitlines in the sub-row are accessed sequentially. After a small amount of time, the sense amplifier that is connected to the bitline in question is selected by raising the CSR signal. This activates the amplifier, producing a current on the data bus determined by the amplifier I/O characteristic. The current for a stored '1' will be much larger than the current for a stored '0', particularly in an array with a low C_b/C_s.

The data bus current passes through a primary bus amplifier, and is then converted to a valid digital level by a data converter located at the edge of the memory core. The process of converting the current to a valid digital level is done by comparing the analog data value to a sampled reference value. The result of that comparison is the reconstructed data, and an SRAM is used to retain the data as necessary for restore and I/O operations, described later in this chapter.

Though this scheme is complex, it makes reading with the 4T sense amplifier possible. It also has the potential to reduce power consumption under specific circumstances. During a read, the bitlines do not make full voltage swings. If a sub-row is read and does not need to be restored (in the event of a read-write operation), then no power is consumed by a useless restore operation.

There is a challenge in using this read scheme that arises from leaving data floating on the bitlines for a short amount of time. The problem is with leakage. Leakage mechanisms in the array cause the voltage left floating on the combined cell and bitline capacitance to slowly decay. This signal degradation must be considered by the designer in two ways. Most importantly, the sub-row length must be kept short enough that the column that leaks for the longest amount of time while the sub-row is active (i.e. the last column to be read) stays within noise tolerances. There is an area penalty for enforcing a maximum length on the sub-rows, because more sub-wordline drivers are required in the core. The designer must make a tradeoff between area penalty and noise margins that suits his or her particular design.

3.3.2.3 Write and Restore Operations

Writing and restoring are essentially the same operation in the proposed architecture. In both cases, data is transferred from an SRAM buffer into the array. The

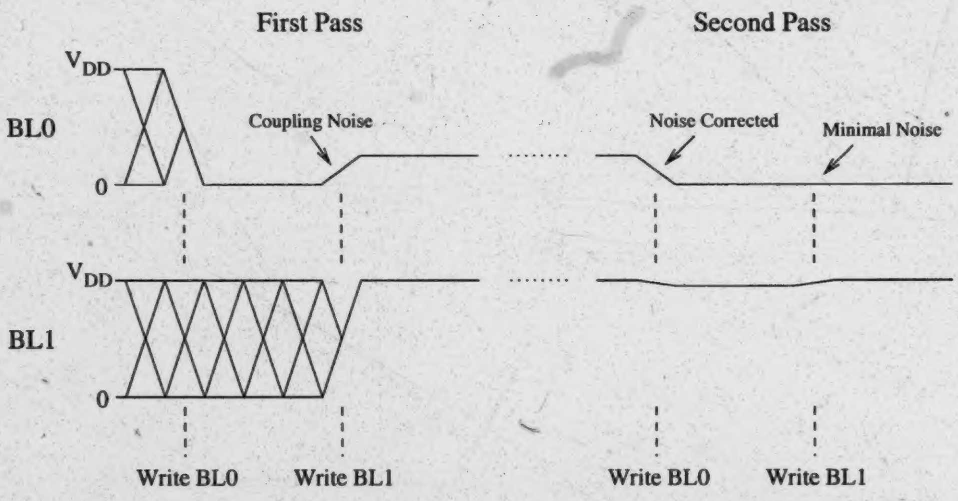


Figure 3.6: Timing Diagram Showing Write Coupling Problem and Solution

only difference is whether the SRAM data was generated from a read operation or by an I/O operation.

For either a write or a restore, the data from the SRAM is written sequentially back into the array by way of a data bus. For a write, a sub-row is first activated; for a restore, the sub-row is already active from the read operation. Data values are written one column at a time. For each column that is written, the data value for that column is placed on the data bus, and then CSW is raised for that column. The data is stored as a dynamic charge on the combined bitline and cell capacitance. CSW is brought low, and the charge remains on the floating bitline until all columns have been written. Once the sub-row of data has been written, the sub-wordline is deactivated and data is stored in the cells.

The practice of leaving bitlines floating while others are written introduces a noise problem in the array. The floating bitlines are susceptible to noise from capacitive coupling to other bitlines that are being actively written. This coupling noise problem is most significant between adjacent bitlines. The "First Pass" timing diagram in figure 3.6 shows how coupling causes noise on one bitline when an adjacent bitline is written and experiences a voltage transition. This noise can substantially reduce the available signal margins for a subsequent read operation.

The solution to the write coupling problem is to make two write passes when writing a sub-row. The first time a row is written, significant coupling noise will appear. This is because some bitlines will make large swings, and the amount of coupling noise on one bitline is proportional to the swing on an adjacent bitline. As shown in the "Second Pass" timing diagram of figure 3.6, bitlines experience much smaller swings on a second write pass, and coupling noise is virtually eliminated. This two-pass scheme is made possible by the relaxed performance requirements of file memory. We are effectively trading off performance for an improvement in noise margins that is necessary to make the proposed architecture feasible. It is worth noting that there is not much additional power consumption for the extra write, because the bitlines experience very small swings on the second pass.

Bitline leakage presents a problem during write or restore, just as it does during read. Since bitlines are left floating while others are written, they are subject to signal degradation due to leakage. Similarly as for read operations, the designer must consider this leakage when determining sub-row length, refresh times, and noise margins.

For this write scheme to be extended to support multilevel memory, only one change is required. A digital-to-analog converter must be used to convert the digital data in the SRAM into analog levels to be stored in the array. Because the write operation is essentially an analog procedure, no other architectural changes would need to be made.

3.3.3 Core Architecture

The ideal core architecture for a DRAM-based file memory maximizes the ratio of array area to support circuitry area, while satisfying noise margin requirements. If sub-arrays are assumed to be as dense as possible, then the goal in developing a core architecture is to use as few sense amplifiers and sub-word decoders as possible. Achieving this goal depends on how bitlines and wordlines are multi-divided, or partitioned, between sense amplifier blocks and between sub-wordline decoders, respectively.

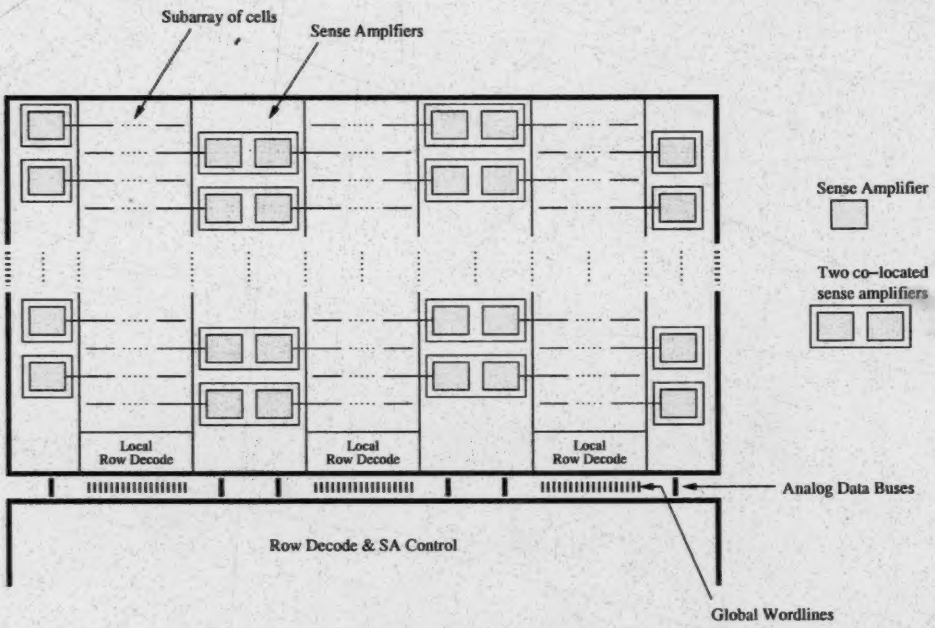

Subarray of cells
Sense Amplfiers
Sense Amplifier
Two co–located sense amplifiers
Local Row Decode
Local Row Decode
Local Row Decode
Analog Data Buses
Row Decode & SA Control
Global Wordlines

Figure 3.7: Sense Amplifier Organization Scheme

Figure 3.7 shows the proposed bitline multi-division scheme for the dynamic file memory architecture. This multi-division scheme allows the sub-arrays to be as dense as possible with an open bitline organization, while also allowing double pitch sense amplifiers. Each bitline connects to a sense amplifier, but each sense amplifier can be laid out in the pitch of two bitlines. This is plenty of space for the minimal sense amplifiers used in the proposed architecture. Sense amplifiers are placed in pairs so that each bitline has its own sense amplifier, but no transistors can be shared between adjacent amplifiers in the four-transistor configuration. For wordline multi-division, a simple scheme that minimizes area is appropriate, performance not being a priority. Wordline multi-division remains necessary, however, so that sub-wordlines can be kept short enough to minimize the effects of leakage during read and write operations, as discussed earlier.

Routing in the core must be done a bit differently in the proposed architecture than it would be done in DRAM. There are fewer signals to route between each subarray because the sense amplifier requires far fewer control signals. However, the

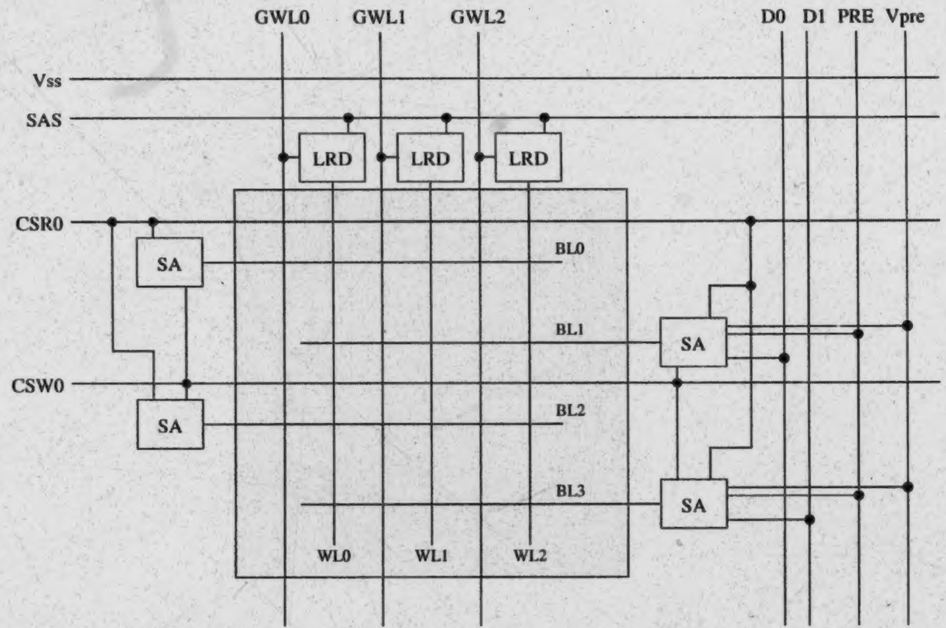


Figure 3.8: Signal Routing in a Sub-Array Unit

need for separate read and write column select lines presents a routing challenge, since the higher levels of metal that are needed to route these signals cannot have as fine a pitch as the bitlines.

Figure 3.8 shows the proposed core routing for a single sub-array. The unit shown is repeated in the horizontal and vertical directions to form a core region. The global wordline (GWL) signals activate local row decoders (LRD) when combined with the sub-array select (SAS) signal. The sense amplifiers are controlled by the precharge (PRE), column select read (CSR), and column select write (CSW) signals. The data buses (D0 and D1) are responsible for transporting data from the sense amplifiers out of the core.

The routing in the figure assumes a DRAM process that supports quadruple-pitch read and write wires. This means that for each sub-array, each CSR and CSW signal connects to two double-pitch sense amplifiers. For this to be possible, two data buses are needed, one for each sense amplifier that shares a single CSR and CSW signal. Otherwise there would be data bus contention between the two sense amplifiers. For any process, the pitch of column select lines should be made as fine as possible, and then the minimum number of data buses required to support those column select lines should be used.

A particular process might have tungsten bitlines and silicided polysilicon wordlines (as in [17]). GWL lines could be routed with the first aluminum layer, along with the data buses and precharge control lines. For the CSR and CSW signal then, the second aluminum layer could be used and these lines could be alternated at that layer's minimum allowed pitch. Alternatively, if a third layer of aluminum is available, then CSR and CSW lines can run parallel on separate layers. Either way, the number of data bus lines needed to support the CSR and CSW lines is equal to the ceiling of the CSR or CSW pitch divided by the sense amplifier pitch.

3.3.4 Memory Architecture

The overall architecture of the proposed file memory varies from that of DRAM in a number of ways. Most of the differences can be attributed to the proposed read/write scheme, but some facilitate techniques that help make the memory denser and less expensive to manufacture.

A block diagram of the overall architecture is shown in figure 3.9, with control circuitry omitted. A four bank memory is depicted, although any number of banks can be used. In the address path, separate row and column addresses are required, or the column address can be eliminated completely for block-only access. To conserve pins on a packaged chip, addresses can be passed in serially rather than as a single word. The row address is decoded and applied to the memory core in the same way as it would be in a DRAM. The column address, if optionally used, follows a different path. Because the memory core only supports serial block access, there is no need to pass a column address into the core. Instead, the column address is used to access the SRAM buffer, which provides data to the synchronous data I/O interface during a read operation. Column select signals in the memory core are controlled by a column address counter, which selects each column one after the other until an entire row is read.

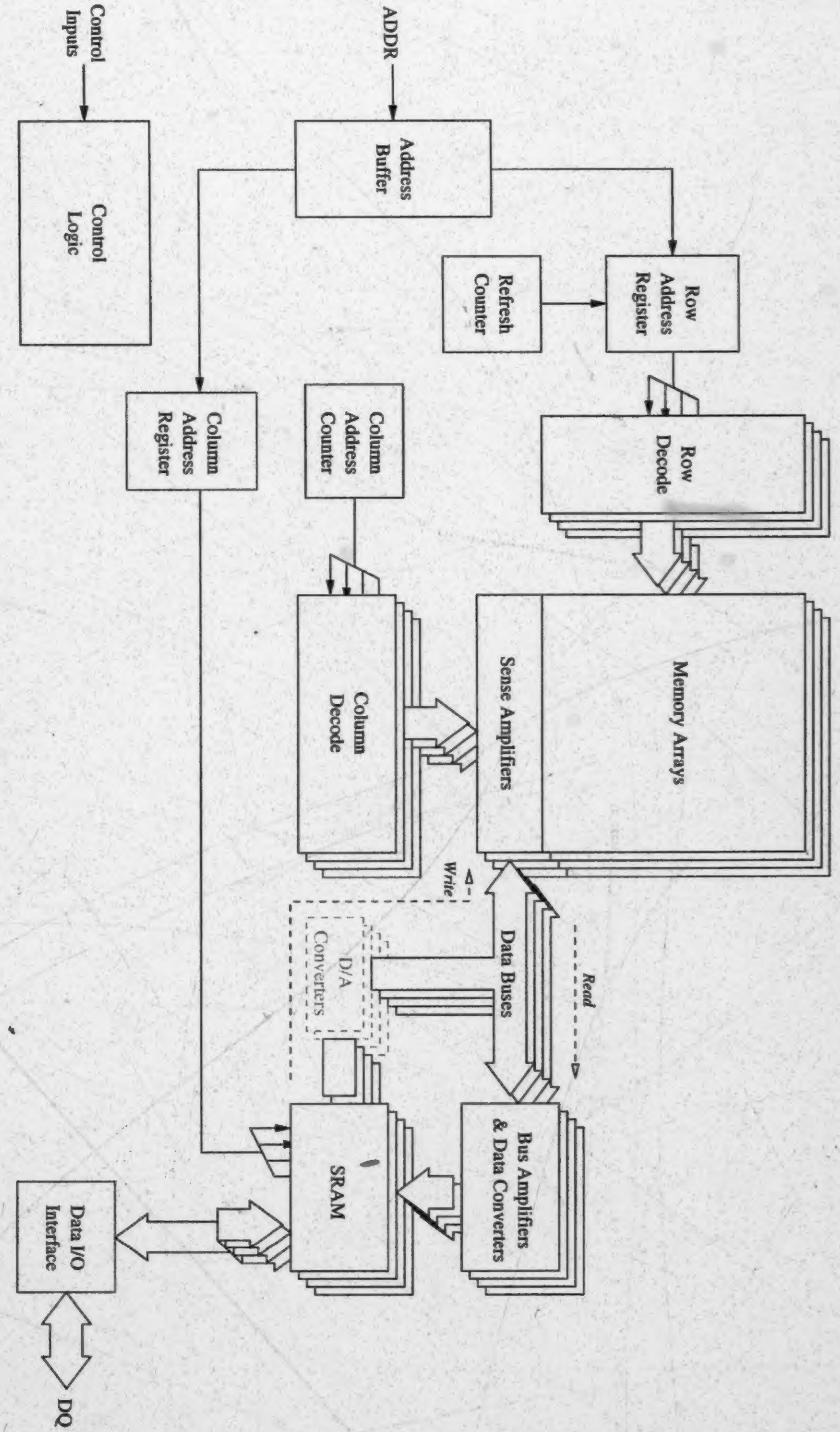

Control Inputs
ADDR
Control Logic
Address Buffer
Row Address Register
Refresh Counter
Column Address Register
Column Address Counter
Row Decode
Memory Arrays
Sense Amplifiers
Column Decode
Write
Read
Data Buses
D/A Converters
Bus Amplifiers & Data Converters
SRAM
Data I/O Interface
DQ

Figure 3.9: Block Diagram

The read datapath begins in the memory banks, where data is sensed and placed on a data bus in the form of a current value. The data is forwarded through a data bus amplifier to a data converter, which converts the analog data to a valid digital data level. Recovered data bits are then stored in SRAM columns for the purposes of I/O and restoration. The write datapath bypasses the data bus amplifier and data converter block to write back to the memory core from the SRAM. For multilevel operation, a digital-to-analog converter (DAC) (shown as a dotted box in figure 3.9) is required to convert digital data into multiple-valued analog data for storage in the core.

3.3.5 Reference Scheme

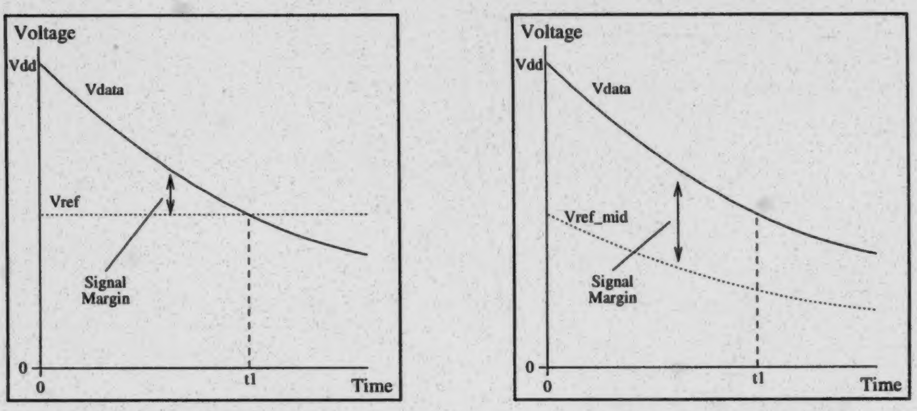
A novel reference scheme has been developed for the proposed architecture that can provide good noise performance and minimize the effects of cell leakage. This reference scheme is possible because of the flexibility gained by placing the analog-to-digital conversion process in the peripheral region rather than local to each sense amplifier (as in conventional DRAMs).

In the proposed reference scheme, extra "reference bitlines" are placed in each sub-array. These reference bitlines contain only reference cells rather than data cells. One reference bitline is required for each valid stored data value; for instance, in a two-level memory there are two reference bitlines. Every time a row is written, the active cells on these extra bitlines are written with the set of valid stored data levels. In a two-level memory then, 0 V would be written to one cell and V_{DD} would be written to the other. These extra writes are always in the same sequence, so the same reference cells always hold the same value. Once a write operation is complete, this set of reference values remains stored in the extra cells, just like regular data.

During a read operation, the reference cells on the active sub-row are read first, in sequence. The current that they effect in the sense amplifier is sampled by a data converter, and held in either analog or digital form for the remainder of the read operation. Once all reference currents have been stored, the actual data in the row is read and converted to digital data. This is achieved by comparing the read data to each sampled reference value; the reference value that most closely matches the read data value represents the reconstructed bit. Note how this differs from conventional DRAMs, which compare stored data to intermediate reference voltages that lie between valid data values.

The great strength of this scheme is its ability to minimize the effects of cell leakage. The concept of how this works is best presented through analogy. Imagine a wine cellar in which you wish to store a large quantity of wine for several decades. You only have a few different types of wine, but you want to store several hundred bottles of each type. Further imagine that you have no easy way of labeling or organizing every single bottle; the only way you can identify bottles once they are in the cellar is by performing a chemical test that compares their contents to those in a known bottle. What makes matters difficult is that the chemical properties of wine change over time. Therefore, if the wine has been in the cellar for a number of years, you can't necessarily identify a bottle by comparing it to a recently made batch of the same type because the chemical properties of the stored wine will have changed. The question then is how can you distinguish a lovely bottle of vintage 1959 Léoville Poyferré from a cheap bottle of your neighbour's secret recipe? The answer is to set aside a small place in the cellar where you can place one bottle of each type, and remember which bottle is which. Since these reference bottles will experience the same climate and aging process as the stored bottles, they can be compared to the stored bottles at some time in the future to determine what type of wine is stored in each bottle. If you are willing to sacrifice a little more space in your cellar, you can leave a set of reference bottles in several locations throughout the cellar. Then, if you compare a stored bottle to the nearest set of reference bottles, you avoid the effects of climate variation between regions of the cellar.

Dynamic memory cells lose their contents gradually over the time between refresh operations. By storing references in cells in the same way data is stored, the references experience the same time-dependent decay as data. Thus, when the data and references are read back out and compared, the data more closely matches the



(a) Conventional Reference

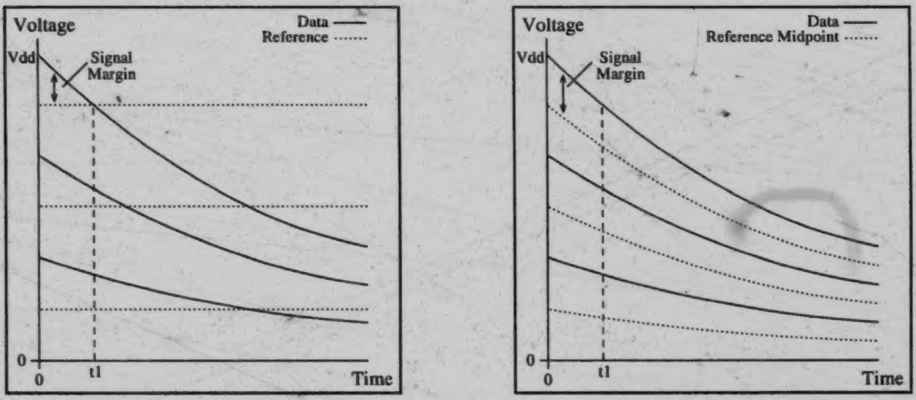
(b) Wine Cellar Technique

Figure 3.10: Signal Margins for Conventional References and Wine Cellar Technique

correct reference. Because of its resemblance to the analogical situation described above, we have coined the term "Wine Cellar Technique" to refer to the proposed reference scheme.

Figure 3.10 shows how the Wine Cellar Technique can improve the signal margins when cell leakage is present. In the figure a two-level memory is assumed. It is also assumed that leakage currents pull cells toward 0 V. In practice, the voltage toward which cells drift is influenced by the combined effect of all cell leakage mechanisms. In particular, subthreshold leakage pulls the cell toward the bitline voltage, PN junction leakage pulls toward the bulk voltage, and gate oxide leakage pulls toward the wordline voltage. The relative weighting of each of these leakage mechanisms, and hence the voltage toward which cells drift, is implementation dependent.

The conventional reference scheme used in DRAMs is shown in figure 3.10(a). As time elapses, the signal margin for sensing a stored V_{DD} level decays rapidly, and at time *t1* the stored voltage has decayed to the point that it cannot be read correctly. In the Wine Cellar Technique depicted in figure 3.10(b), the stored voltage decays as it does it the conventional scheme. However, the stored V_{DD} reference



(a) Conventional Reference (b) Wine Cellar Technique

Figure 3.11: Wine Cellar Technique with Multilevel Memory

value also decays, ideally at the same rate as the stored data value. The dotted line labeled V_{ref_mid} shows the midpoint between the two stored references (0 and V_{DD}) as each of them drifts. This midpoint represents the decision threshold between a "high" and a "low" data level.

The Wine Cellar Technique is also very effective in a multilevel memory. As shown in figure 3.11, signal margins are larger for a much longer period of time when using the Wine Cellar Technique with multiple stored levels. Keep in mind that the reference curves in figure 3.11 are not actual stored values; rather, they are the midpoint between two stored reference values[2].

Another advantage of the Wine Cellar Technique is that no intermediate reference voltages need to be generated anywhere in the memory. The reference voltages are equal to the data voltages. This does not help in the case of two-level memory, since a precharge voltage must be generated anyway, but in multilevel memories this can be a significant advantage.

There are some challenges associated with using the Wine Cellar Technique. For the technique to be effective, all cells must experience similar amounts of leakage. This means that process control must be very strict so that variations in leakage between cells is within a tolerable range. Otherwise, noise margins can be substantially degraded. Process control must also be very strict for the sense amplifiers. Because the stored references are read with a sense amplifier in the same way as data, significant variation in a reference sense amplifier will cause a reduction in noise margins. At device testing, if variation tolerances are not met, then redundant rows or columns would need to be swapped in to replace offending cells and sense amplifiers.

---

[2] It is possible to store actual midpoint values rather than storing the same values as data for references. Little architectural modification is needed to support this, but additional voltage generators are required in the periphery to supply the midpoint voltages.

Another requirement needed to ensure that the cells experience similar amounts of leakage is that inactive bitlines must all be precharged to the same level at all times, otherwise subthreshold leakage will vary between cells. This is standard practise in most commercial DRAMs. The control circuitry must make sure that if some bitlines must float for a certain period of time during inactivity, they float for as short a time as possible.

A small area penalty is paid to implement the Wine Cellar Technique, due to the need to have one extra column per data level. This penalty is not major, especially when comparing a memory with the Wine Cellar Technique to a conventional DRAM with dummy wordlines. Dummy wordlines are often included in DRAM designs to improve symmetry between data and reference bitlines, but they are not beneficial in the absence of differential sense amplifiers. If a 1-Gb, two-level memory with 512 sub-wordlines and 256 bitlines per sub-array is considered, then the number of reference cells required to implement the Wine Cellar Technique is equal to the number of dummy cells required in a conventional implementation. From this perspective, there is essentially no area penalty for adding reference bitlines to a two-level memory. For a multilevel memory, the area penalty becomes insignificant due to the substantial density improvement offered by multilevel storage.

3.3.6 Data Bus Amplifiers

During a read operation, the data bus amplifiers are responsible for driving the common data buses that are shared within blocks of sense amplifiers. Data buses require specific conditions to allow the sense amplifiers to properly drive them, and these conditions must be provided by data bus amplifiers. A constant voltage must be held on the data bus so that each sense amplifier experiences the same bias conditions, and sufficient current must be supplied to each data bus. The data bus amplifiers must generate these conditions, and prepare data for use as the input to a data converter. When a read operation begins and a sense amplifier has been selected to drive a data bus, the data bus amplifier must transfer the analog bus current to a data converter. One amplifier is required per data bus. Two suitable bus amplifier circuits are presented in this section.

A current mirror amplifier circuit, as shown in figure 3.12, provides appropriate current and voltage conditions on the data bus. It also isolates the data bus from the output driver so that the input load of subsequent circuitry is unimportant. The BUS_PRE signal is used before each amplifier operation to pull the bus to V_{DD}. The PMOS current mirror can not pull strongly to V_{DD} on its own, so precharging the bus substantially reduces worst case sensing time. The V_{bias} transistor is optional, its inclusion dependent on the data converter implementation. This transistor is a simple way to convert the amplifier output current to an output voltage.

Exact matching of the two current mirror transistors is not critical because having an exact copy of the data bus current is not absolutely necessary. In fact, process variation of any kind has very little effect on the amplifier. As long as the amplifier output is monotonic and consistent from one read to another (as is the case for the current mirror), the analog data can be amplified and sensed reliably. The PMOS transistors should be sized as large as area requirements allow so that read time is minimized. The bias transistor, if used, is sized to produce a suitable output voltage.

A bus precharge amplifier, as shown in figure 3.13, also provides appropriate current and voltage conditions on the data bus. This circuit uses a simple amplification scheme that precharges the bus before each read. When a bitline sense amplifier is activated, the amplifier transistor sinks current from the bus at a rate determined by the bitline voltage. If the column select signal is pulsed for a fixed length of time, then the final bus voltage will be proportional to the bitline voltage.

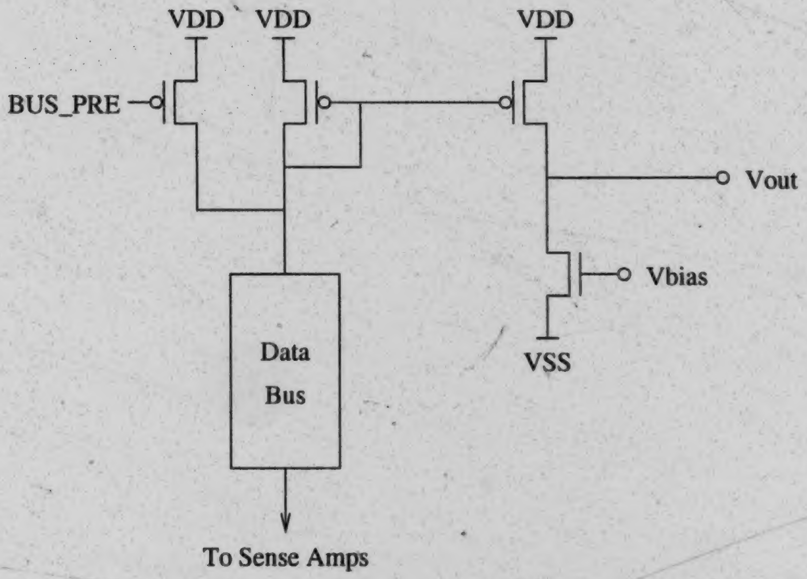

VDD
VDD
VDD
BUS_PRE
Vout
Vbias
VSS
Data
Bus
To Sense Amps

Figure 3.12: Current Mirror Bus Amplifier

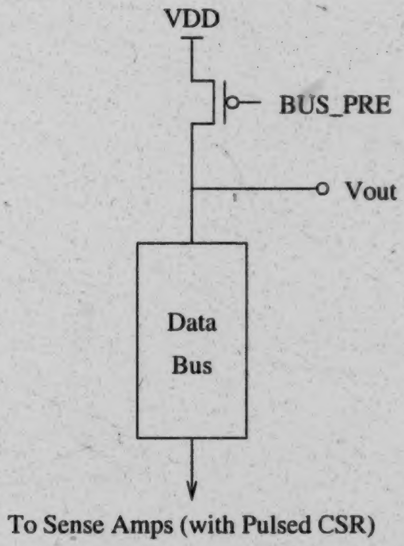

VDD
BUS_PRE
Vout
Data
Bus
To Sense Amps (with Pulsed CSR)

Figure 3.13: Bus Precharge Amplifier

The PMOS precharge transistor should be sized as large as area requirements allow, to minimize read time. However, the read time with this amplifier configuration is relatively slow, and power consumption is quite large because the bus is precharged and discharged for each read operation. Also, tuning a subsequent data converter input to the bus capacitance can be difficult in the face of process variation. Nonetheless, the simplicity and small size of this configuration could make it useful in some implementations.

3.3.7 Data Converters

The focus of the following discussion is on the architectural aspects of the data conversion units, and their influence on core operation. A variety of circuit configurations can be chosen for these units, but the details of their implementation are not specified in this work.

Data conversion circuitry is required to translate the analog read signal output of a bus amplifier into a valid digital logic level. This is essentially a simple analog-to-digital conversion process. A converter must sample the reference values when they are first read from the array, and then compare subsequent data values to the references to determine the logic level to which those values correspond.

In a multilevel memory, data conversion would occur in the same fashion, except more references would be stored, and more comparisons would be required. The data converters could accommodate these requirements either with parallelism or time-multiplexing of the conversion operation. Additionally, a digital-to-analog converter would be needed for write and restore operations.

One data converter unit is required for every data bus that will be active at one time in an implementation of the proposed architecture. Because of the proposed architecture uses a serial read scheme, an implementation will require on the order of hundreds of data converters in order to meet refresh requirements; the number of converters required for a specific 1-Gb implementation is discussed in section 4.8. The design of the data converters can be relatively simple[3], and therefore the area penalty of implementing a few hundred of them is not prohibitive.

---

[3] For instance, the analog-to-digital and comparison process could be done by charging capacitors

3.3.8 SRAM Buffer

The final component of the architecture that merits discussion is the buffer used to store data before write operations and after read operations. SRAM is a very good technology for this buffer because of its relatively high density, which minimizes its area requirement, and its good performance, which allows a high external I/O rate. However, other types of memory could be used. The buffer only needs to be a type of memory that can outperform the file memory core, and it should take up as little chip area as possible.

Each SRAM unit must have sufficient capacity to hold a sub-row of data, and one unit is required per data converter. The SRAM needs to have sequential bit access for read, write, and I/O operations. It also may be desirable to use a dual-ported SRAM to allow I/O operations simultaneously with restore operations. Regardless, a basic SRAM is sufficient for successful operation of the file memory.

3.4 Summary

The proposed architecture for an economical, DRAM-based semiconductor file memory attempts to reduce production costs by increasing density and reducing the number of address pins required. An open bitline array organization reduces array area, and a four-transistor sense amplifier reduces the area required between arrays. The number of address pins is reduced through the architectural support for serial address transfer.

For coping with potential noise problems introduced by the use of an analog sense amplifier and an open bitline array, a number of techniques are proposed. Signal margins can be improved by reducing bitline length, thereby reducing the extent of signal dilution from charge sharing during a read operation. This tradeoff is acceptable because additional sense amplifiers require very little area. Coupling

---

(which are easily designed in a DRAM process) with an output current from the bus amplifiers. The capacitor voltages could then be compared with one or two comparators.

noise is reduced with a two-pass write operation. Finally, leakage effects are minimized by storing references along with data in the "Wine Cellar Technique."

Despite the use of several specialized techniques, there are still a number of challenges for implementing the proposed architecture. The architecture is quite sensitive to process variations between devices that are spatially nearby one another. The architecture is designed to be insensitive to global variation, but if multiple devices in the same memory sub-array exhibit substantial variation, a large reduction in noise margins can occur. Another challenge is to select sub-wordline lengths that are appropriate. Sub-wordlines that are too long will allow too much leakage during read and write; sub-wordlines that are too short will reduce the density of the chip. The optimal tradeoffs between line lengths and noise margins will depend on the chosen implementation technology.

A strength of this architecture is its potential for multilevel storage. Little architectural overhead is required for the architecture to support multilevel data, so the possible increase in density is large.

Chapter 4 analyzes and quantifies the important aspects of this architecture, and evaluates how well the architecture satisfies the requirements of a DRAM-based file memory design.

Chapter 4

Analysis and Simulation Results

4.1 Overview

For the proposed dynamic file memory architecture to be feasibly implemented as a semiconductor integrated circuit it must satisfy the requirements outlined in section 3.2.2. Furthermore, the circuits and techniques employed must be characterized so that design tradeoffs can be fully understood. This verification and characterization is achieved through detailed analysis and simulation of the architecture and its constituent circuits. To facilitate interpretation, the proposed architecture is compared to published DRAM designs in terms of area, functionality, performance, and power consumption.

This chapter focuses specifically on a one bit per cell implementation of the proposed architecture. Many of the results that are found here reveal valuable information about multilevel implementation as well; chapter 5 examines the intricacies of multilevel implementation in greater detail.

4.2 Method

Two primary techniques are used to evaluate the proposed architecture. The first technique is theoretical analysis, in which the proposed architecture is modeled with equations that describe its characteristics (both absolute and relative to a standard DRAM). The second technique is detailed simulation, in which computer simulation is used to determine the exact behaviour of various circuits.

Theoretical analysis is performed using process parameters from the United Microelectronics Corporation's (UMC's) 0.13-μm Mixed Mode CMOS fabrication process [46, 45, 44], which features a V_{DD} of 1.2 V. This process is selected for lack of a complete process parameter set for any particular DRAM process. Every effort is made, however, to ensure that results are applicable to integrated circuits fabricated in a DRAM process, and to ensure that any relationship between stated results and results to be expected in a DRAM process is made clear. In some circumstances, assumed DRAM parameters are substituted for the Mixed Mode CMOS parameters. Such substitutions are also clearly indicated. Whenever appropriate, theoretical results are confirmed through simulation. The Mathworks' Matlab is used in some cases to generate theoretical data sets suitable for plotting, based on the equations and models presented.

Simulations are also performed using the United Microelectronics Corporation's 0.13-μm Mixed Mode process parameters. Circuit simulations are prepared using the Cadence Analog Artist software in combination with Cadence's Virtuoso Schematic tool. Analog simulations are performed using Avant! Corporation's Star-Hspice release 2001.4, with the ACCURA and KCLTEST flags set and with an ABSV (VNTOL) value of 10^{-15} to ensure sufficient accuracy in cell array simulations. All other options are left at their default values. Transistor SPICE models used are level 49 (BSIM3v3) MOSFET models provided by UMC. BSIM3 models include thorough modeling of almost all MOSFET characteristics relevant to DRAM circuits. The only disadvantage of using a BSIM3 model for DRAM simulations (compared to a BSIM4 model, for instance) is that some very small geometry effects, such as gate oxide tunneling, are not considered by the BSIM3 model [3]. For a 0.13-μm process, the error introduced by ignoring these effects is very small; however, for smaller geometries the BSIM3 model would likely provide inaccurate results in some simulations.

Analog stimuli and testbenches are generated with the WaveGen toolset, which was developed by Tyler Brandon at the University of Alberta. Functional hardware description language simulations are performed using the Cadence Spectre simulator version 5.0.

A detailed description of the parameters used for analog memory core simulations is included in appendix B.

4.3 Density Improvement Analysis

Before characterizing the operation of the proposed file memory, the improvement in density of this architecture over a conventional DRAM architecture is considered. For the proposed architecture to be economical, it must be denser than conventional DRAM.

A typical DRAM designed in a process with three levels of metalization, trench capacitors, hierarchical wordlines, and a folded bitline array is analyzed for its density. The proposed architecture is then analyzed and its density is compared to that of the typical DRAM to determine the amount of improvement.

A sub-array of cells without any decode or sensing circuitry has an area given by

$$A_{array} = (N_b P_b)(N_w P_w) \, , \tag{4.1}$$

where N_b and N_w are the number of bitlines and wordlines in a sub-array, respectively, and P_b and P_w are the bitline and wordline pitches, respectively.

A core unit, defined as a basic sub-array with supporting sense amplifier and local row decode strips, has an area of

$$A_{core\_unit} = (N_b P_b + H_{lrd})(N_w P_w + W_{sa}) \, , \tag{4.2}$$

where H_{lrd} is the height (in the direction of the wordlines) of a local row decoder, and W_{sa} is the width (in the direction of the bitlines) of a sense amplifier, including I/O routing and column access devices. Also, any area occupied by bitline twist regions is included in the sense amplifier width value.

The area of the memory chip is then

$$A_{chip} = N_{banks} N_{core\_units/bank} A_{core} + A_{periphery} \, , \tag{4.3}$$

where N_{banks} is the number of banks in the chip, $N_{core\_units/bank}$ is the number of core units per bank, and $A_{periphery}$ is the area occupied by peripheral circuitry, including the main row and column decode blocks, sensing control circuitry, I/O pads, and other circuits for tasks such as synchronization and voltage conversion. This equation ignores array redundancy, which is assumed to have a limited influence on the overall area comparison between conventional DRAM and the proposed architecture.

The above equations hold true for both conventional DRAM and the proposed architecture; however, the values of the variables are different for each. Because the proposed architecture uses an open bitline array organization that supports cross-point cells, the number of wordlines per sub-array is halved while the wordline pitch is increased by 1.5 [16]. The width of the sense amplifier regions is greatly reduced due to the novel sense amplifier design employed, and through the elimination of twist regions. The periphery of the proposed architecture will require a larger area due to the need for data converters and SRAM cells.

Based on [16, 19, 42], a typical 1-Gb DRAM with folded bitlines has $8F^2$ memory cells, a $2F$ bitline pitch ($4F$ bitline pairs) and a $2F$ wordline pitch, 512 cells per sub-wordline and 256 cells per bitline. It also has a sense amplifier and twist region width of $210F$, and local row decode height of $123.3F$. Periphery overhead is typically around 20%. Under these conditions, such a DRAM has an area of $13.72\text{x}10^9F^2$.

A 1-Gb, two-level-storage file memory chip using the proposed architecture has $6F^2$ memory cells in an open bitline organization [16] (as shown in figure 3.7), a $2F$ bitline pitch and a $3F$ wordline pitch, with 512 cells per sub-wordline and 256 cells per bitline. It is assumed that the periphery will occupy the same area as it does in a typical DRAM, plus the overhead of SRAM. Much of the same circuitry is required in the proposed file memory, and the small area of new circuitry (such as data converters) can be offset by the elimination of some high-performance synchronization and I/O circuitry, so this is a reasonable assumption. The area of SRAM, with $2.5\text{-}\mu\text{m}^2$ cells (based on [34]) and 30% control overhead, is $0.13\text{x}10^9F^2$ per Mb.

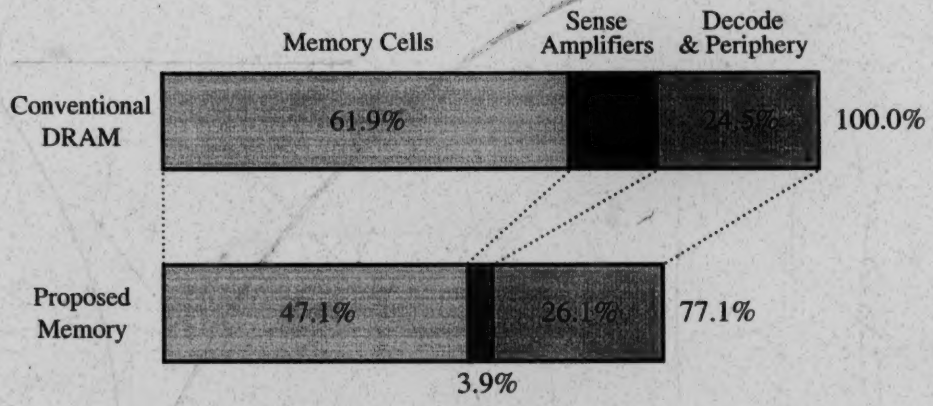


Figure 4.1: Area Comparison of Two Memories with 2^{30} Dynamic Cells

Approximately 512 kb of SRAM is necessary, with an area of $0.07\text{x}10^9F^2$. Sense amplifier regions have a width of $56F$, based on the 0.13-μm implementation design rules, and local row decode regions have a height of $123.3F$ as in the typical DRAM described above. Under these conditions, a 1-Gb implementation of the proposed architecture has an area of $10.56\text{x}10^9F^2$.

Given these values, the proposed architecture requires 77.0% of the area of a DRAM with equivalent capacity. The reduction in each of the major contributors to the overall area is shown in figure 4.1. It is apparent that the area reduction is primarily due to the open bitline array organization and the reduction in sense amplifier area. The area overhead of reference bitlines is almost insignificant at 0.2%, and the area overhead of SRAM is also very small at 0.5%.

Because of the small area of sense amplifiers in the proposed architecture and the reduced noise immunity of an open bitline array organization, it is important to consider the area tradeoff if bitline length is reduced for the sake of improving noise margins. Table 4.1 shows the change in area and area ratio for the proposed architecture with varying bitline lengths for a sub-wordline length of 512 cells, assuming overall memory capacity stays constant. It is worthwhile to note that even very short bitlines allow an area improvement over a typical DRAM.

Table 4.1: Area of the Proposed Architecture for Different Bitline Lengths

Cells per FM Bitline	A_{fm} (x10^9F^2)	$\frac{A_{fm}}{A_{dram}}$	$A_{sense_{fm}}$ (x10^9F^2)	$\frac{A_{sense_{fm}}}{A_{sense_{dram}}}$
256	10.56	77.1%	0.53	28.3%
128	11.09	80.8%	1.05	56.2%
64	12.15	88.6%	2.11	113.0%

4.4 Functional Simulation

A simple behavioural model of the core of the proposed architecture is developed to allow functional verification of the design. The model is created with the "Verilog-A" hardware description language, based on a subset of the architectural diagram shown in figure 3.9. Verilog-A was chosen because the array, sense amplifiers, buses, data bus amplifiers, and data converters are analog components, and modeling them with a purely digital language would oversimplify the simulation and compromise the validity of results. All of the code used for functional simulation can be found in appendix E.

Figure 4.2 shows a write operation followed by a read operation for four cells: two reference cells and two data cells. The first reference cell is written with a '0', and the second is written with a '1'. The data cells are opposite, written with a '1' first and a '0' second. Note that the timing shown in the figure is arbitrary. The writes occur at 10, 20, 30, and 40 ns in the simulation, indicated by the sequential rising and falling of the four column select write (CSW) signals. The subsequent read begins at 75 ns with wordline activation. As the four column select read (CSR) signals are asserted, the voltage output of the data bus sense amplifier for the row being sensed changes ("Output Bus Voltage" in the figure). The correct data output is observed in the output signal with a '1' being read at 100 ns and a '0' being read at 110 ns. This simple result verifies that the fundamental aspects of the proposed architecture function as intended.

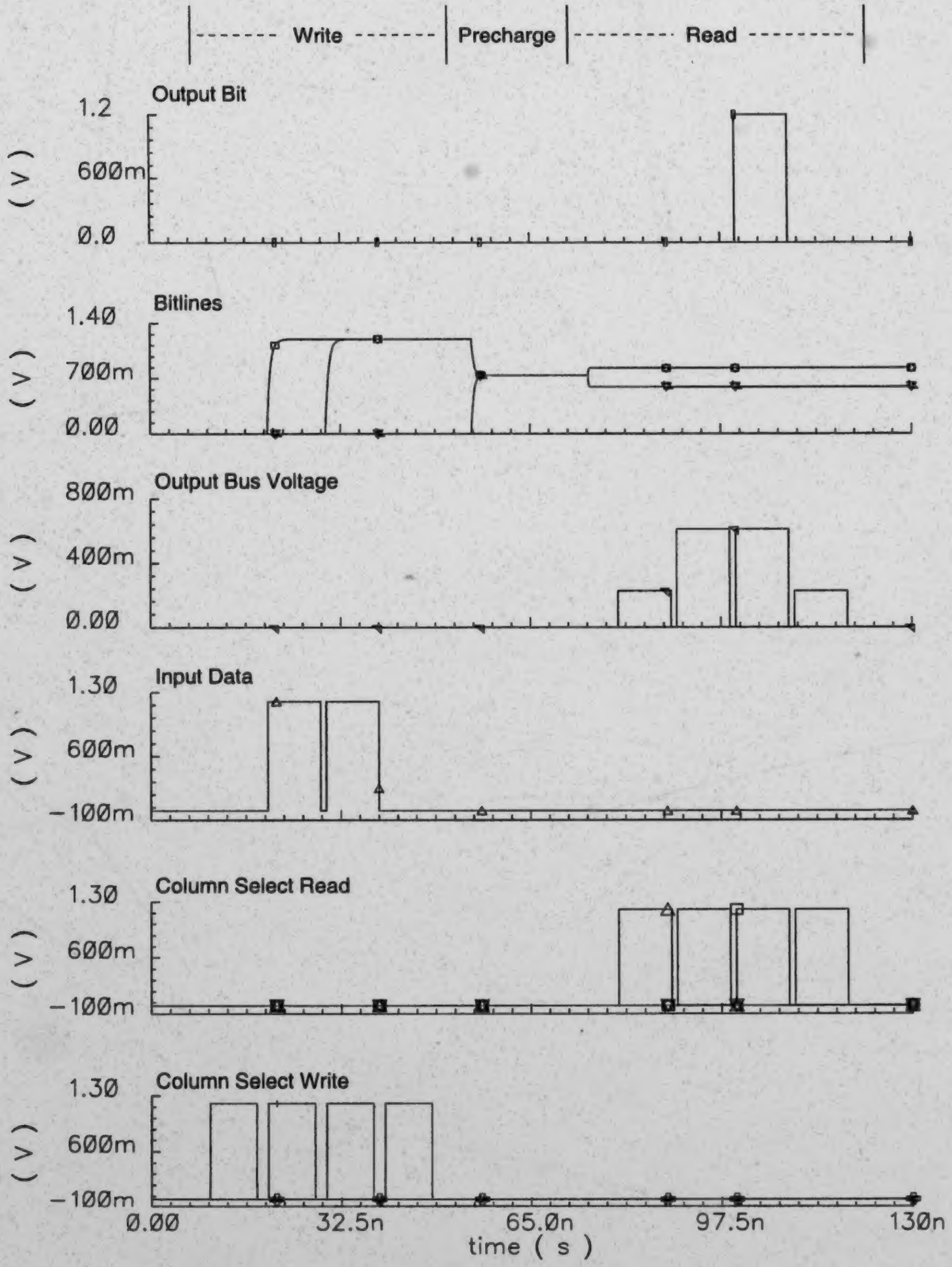


Figure 4.2: Functional Simulation Results

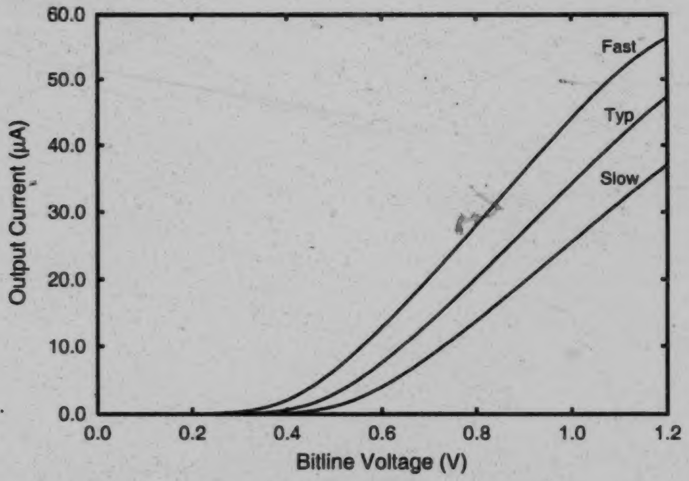


Figure 4.3: Sense Amplifier I/O Characteristics for Typical, Fast and Slow Models

4.5 Sense Amplifier Simulation

The sense amplifier design is simulated with a bus model that assumes an 8-mm long bus, which is approximately the worst case length of a bus in a 1-Gb, 0.13-μm implementation of the proposed architecture. The precharge and write transistors are minimum sized, while the sense and select transistors are sized at double minimum width and length to reduce the effects of process variation.

The I/O characteristic for the amplifier is shown in figure 4.3 for typical, fast, and slow transistor models.

Examining the I/O characteristics, it is apparent that the amplifier has approximately constant transconductance over the range V_{th} to V_{DD}, The average gain over the amplifier's operating region is 58.1 μA/V in the typical model, with a range of 45.9 μA/V to 67.8 μA/V using slow and fast models, respectively.

4.6 Bitline Biasing

As discussed in chapter 3, any precharge voltage can legitimately be chosen to bias the sense amplifiers. However, certain bias voltages will lead to better noise margins. The goal when precharging the bitlines is to bias the sense amplifier so as to provide the largest noise margins while keeping the sense amplifier within its valid operating range under worst-case conditions.

Optimal biasing is accomplished by biasing the sense amplifiers as close to $V_{DD}/2$ as possible, while ensuring that the bitline voltage stays within the valid operating region of the sense amplifiers. It is necessary to bias the amplifiers as close to $V_{DD}/2$ as possible because of capacitive coupling within the array. As the precharge moves farther away from $V_{DD}/2$, the bitline swing becomes much larger for reading one value and smaller for the other (i.e. more swing for a '0' and less for a '1'). Since the reference value stays fixed, there will be a much larger signal degradation for a small-swing bitline that is adjacent to several large-swing bitlines than there would be in the converse case. This leads to a problematic asymmetry in worst case noise margins.

The optimal bias point is dependent on the sense amplifier I/O characteristic and the bitline capacitance. To simplify the analysis, it is assumed that the sense amplifier is linear in its valid operating region. While this is not completely true, the effects of non-linearity are relatively small, and a bias point found under this assumption will still be very good. It is also assumed in this analysis that the valid input operating region of the amplifier is mostly above $V_{DD}/2$, as is the case in the results of section 4.5.

To find the optimal bias point, first the cell ratio (the ratio of bitline capacitance to cell capacitance) of the array is determined. The more accurately the cell ratio can be determined, the better the noise performance that can be achieved. Next, the lower bound for the valid operating region of an amplifier is found based on amplifier simulation. The lower bound must be the worst case lower bound (i.e. the lower bound for a slow amplifier model at the lowest allowable operating temperature), so that the amplifier will still perform well under these conditions. Finally, it is verified that the bitline signal for the given cell ratio is well within the valid input range of the amplifier. If the bitline signal is not primarily within that range, then the amplifier is not being used optimally and the bias should be increased.

Table 4.2: Capacitance and Bias Voltage for Different Bitline Lengths

Cells per Bitline	Cell Ratio	Bias Voltage (V)	Valid Signal Size?
32	0.58	1.369	No
64	1.08	0.962	Yes
96	1.59	0.815	Yes
128	2.09	0.739	Yes
160	2.60	0.692	Yes
192	3.10	0.661	Yes
224	3.61	0.638	Yes
256	4.12	0.621	Yes
288	4.62	0.608	Yes
320	5.13	0.600	Yes
352	5.63	0.600	Yes
384	6.14	0.600	Yes
416	6.64	0.600	Yes
448	7.15	0.600	Yes
480	7.66	0.600	Yes
512	8.16	0.600	Yes

Given the values of cell ratio and lower bound, the optimal bias point is

$$V_{bias_{opt}} = \max\left\{V_{lower\_bound} \cdot \left(1 + \frac{C_s}{C_b}\right), \frac{V_{DD}}{2}\right\}, \tag{4.4}$$

where V_{lower_bound} is the lower bound of the sense amplifier input range, and C_s and C_b are the cell and bitline capacitance, respectively, as defined in chapter 2. This equation calculates the bias point by setting the lowest possible bitline voltage during a read operation equal to the sense amplifier's lower bound. The bias point is the precharge voltage that will produce the lower bound voltage when a '0' is read, unless that voltage is lower than $V_{DD}/2$, in which case a precharge of $V_{DD}/2$ is used.

Table 4.2 shows the theoretically determined cell ratio and calculated bias point for the 0.13-μm implementation that is being considered in this chapter, with a cell capacitance of 25 fF.

The data in table 4.2 show the bias voltage decreasing as the bitline length increases. This reflects the decrease in signal size as bitline capacitance increases. The bias voltages calculated here are used for the remaining simulations and analyses in this chapter unless otherwise noted.

4.7 Noise Margins

It is crucial that the noise margins in the proposed architecture are carefully examined under all possible operating scenarios. An implementation of the proposed architecture will not function correctly if the noise margins are not satisfied, rendering that implementation useless. This section presents noise simulation results for the 0.13-μm implementation being considered in this chapter.

To make the measurement of noise margins tractable, and the results more understandable, noise margin simulations are divided into three categories. These are: noise due to parasitic capacitive coupling, noise due to process variations and offset, and the reduction of noise margins due to leakage during read, write, and restore operations. The combined effect of these noise contributors is then considered at the end of this section.

4.7.1 Parasitic Capacitive Coupling Noise

The noise contribution from parasitic capacitive coupling is simulated for a write followed by a read using two worst case data sequences on a group of five adjacent bitlines. These worst case data sequences are "00100" and "11011," where each digit in a sequence represents the data being written to a different bitline in the group of five bitlines. Each of these sequences has the capability to cause significant degradation in the middle signal of the sequence because the opposing adjacent values couple on to the center bitline.

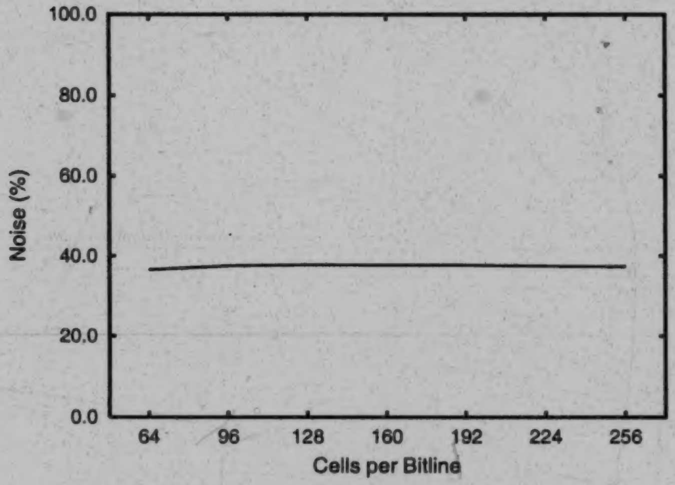
Simulations are performed by writing one of the two data sequences to cells on five adjacent bitlines along a single wordline. Following this write, the wordline is deactivated and the bitlines are precharged. The wordline is then reactivated, and readings are taken for the resulting bitline voltages and sense currents coming from the sense amplifiers. Results are gathered for various bitline lengths using typical, slow, and fast process corners.

The worst case coupling noise occurs for the pattern "00100." There are two reasons for this. The first is that the bias point is above $V_{DD}/2$ for many of the bitline lengths. Therefore, there is a larger voltage swing on a bitline when a '0' is read than when a '1' is read. This larger swing causes greater noise on the adjacent '1' bitline. The other reason is that the gain of the amplifiers is larger at larger input voltages. At larger values of gain, noise is amplified more, so the higher input voltage of the '1' value experiences greater noise.

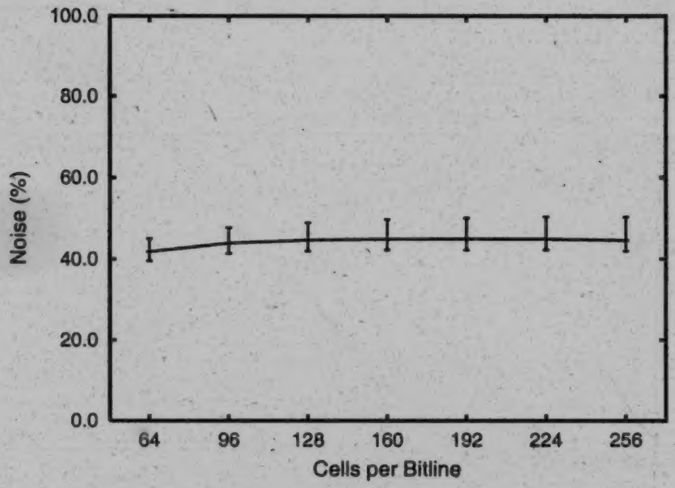
Figure 4.4(a) shows the worst case coupling noise voltage versus the number of cells per bitline, as a percentage of the overall signal, for different bitline lengths. Figure 4.4(b) shows the worst case coupling noise at the output of the sense amplifier, again as a percentage of the overall signal, for the same configurations. The vertical error bars in figure 4.4(b) show the range of possible values due to process variations.

The results in figure 4.4 are obtained by first simulating the memory core in the absence of coupling, with adjacent bitlines tied to a fixed voltage. Bitline voltage and sense amplifier current during a read operation are measured for different bitline lengths using slow, typical, and fast process corners at 0 ^oC and 80 ^oC. The memory core is then simulated with coupling, and bitline voltage and sense amplifier current are again measured. With this data, coupling noise is calculated as the worst case difference between expected and measured read values. For coupling noise, the worst case conditions occur with slow process models at 80 ^oC. In figure 4.4 the coupling noise is given as a percentage of the read signal, which is the difference between an expected '1' or '0' voltage and the reference voltage.

Figure 4.4 indicates that coupling noise as a percentage of the read signal remains approximately constant; however, the size of remaining noise margins is reduced as bitline length increases, as shown in figure 4.5. This reflects the fact that there is a maximum bitline length to achieve adequate noise margins. This maximum length is determined later in this section, once all noise data has been examined. As before, the vertical bars in figure 4.5 show the range of values due to process variations.



(a) Noise at Bitline



(b) Noise at Sense Amplifier Output

Figure 4.4: Worst Case Noise due to Capacitive Coupling for Different Bitline Sizes

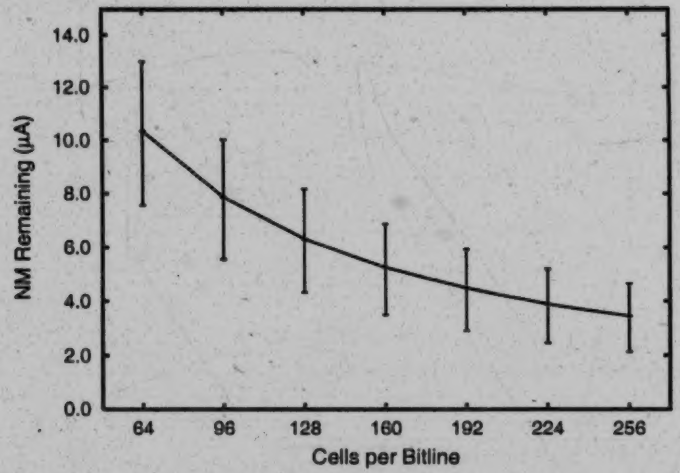


Figure 4.5: Noise Margins Remaining After Capacitive Coupling for Different Bitline Sizes

4.7.2 Process Variations and Offset Noise

The core circuitry is simulated in the absence of coupling noise and leakage effects, using typical, slow and fast models, to isolate the effects of process variation on noise margins from the effects of other noise sources during read and write operations.

There are two different ways that process variations can reduce noise margins during sensing. The first is when all of the sense amplifiers in a single core unit are nearly identical, but they deviate from typical characteristics. This is referred to as "global" process variation, because all circuits are affected equally. The second possibility is when sense amplifiers on a single bus differ from one another. This is referred to as "local" process variation, because the characteristics of each circuit within a spatial region can be different.

Figure 4.6 shows the worst case "noise" introduced through each type of process variation. The term "noise" in this context refers to the amount by which noise margins are reduced due to process variations. For local variations, the worst case noise is determined differently for a '0' than for a '1'. For a '0', a combination of a "slow" reference sense amplifier and a "fast" data sense amplifier is used, while

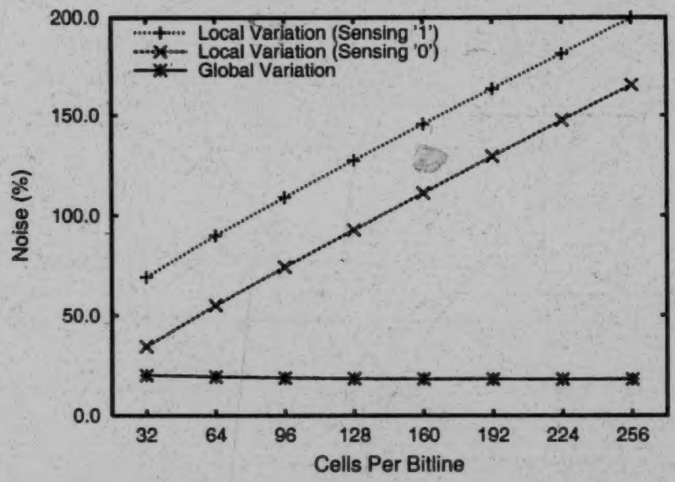


Figure 4.6: Noise due to Process Variations For Different Bitline Sizes

for a '1', a "fast" reference amplifier is used with a "slow" data amplifier. The reduction in noise margins is given by

$$Noise_0 = (ref_{typical} - 0_{typical}) - (ref_{slow} - 0_{fast}) \tag{4.5}$$

and

$$Noise_1 = (1_{typical} - ref_{typical}) - (1_{slow} - ref_{fast})\,. \tag{4.6}$$

Curves are shown in figure 4.6 for local variations when a '0' is sensed and when a '1' is sensed, and for global process variations when sensing either a '0' or a '1'. The noise due to local process variations is different for a '0' and a '1' because the difference between typical, slow, and fast devices varies at different points on the output characteristic. On the other hand, the noise due to global process variations is the same for a '0' and a '1' because only one type of device variation (typical, slow, or fast) affects the noise margins at a time.

The worst case noise introduced by global process variations is relatively small, hovering around 20% for all bitline lengths. Global process variations reduce noise margins by reducing the gain of sensing transistors (in the case of slow transistors). If noise margins are designed for a typical transistor model, then a slow transistor will cause a reduction in output signal by approximately 20%. The trend for noise to remain approximately constant for all bitline lengths reflects the proposed architecture's ability to tolerate global process variations.

The worst case noise introduced by local process variations is very substantial according to figure 4.6. However, this noise data assumes that worst case local variation is equivalent to the maximum possible variation between any devices in the 0.13-μm technology, even if they come from different wafer lots. Such an assumption results in extremely conservative data that ignores the spatial correlation of devices on a chip and even correlations between dies on the same wafer. This assumption is necessary due to a lack of available information about the statistical properties of both the 0.13-μm technology being considered and DRAM processes in general. Despite the extent to which the noise results in figure 4.6 are unrealistic, they still support the concept that careful process control is imperative; spatially co-located transistors must be as closely matched as possible, as mentioned in section 3.3.5.

4.7.3 Effect of Leakage on Noise Margins

Leakage degrades the noise margins in two ways. The first is through the conventional cell leakage that affects all DRAMs. The second is leakage from the floating bitlines during read and write operations. Conventional cell leakage is not considered in detail here, as it has the same effect in the proposed architecture as it does in DRAMs. For subsequent analysis, it is assumed that the worst case cell leakage results in a 10% signal degradation in 256 ms, a standard refresh interval. Unlike conventional DRAM, leakage from bitlines during read and write has an effect on noise margins in the proposed architecture. As described in section 3.3.2, bitlines are left floating during read and write operations, leaving them susceptible to charge leakage.

The exact magnitude of cell and bitline leakage varies greatly from one technology to another. Models for the 0.13-μm CMOS technology being used in this chapter give unrealistically pessimistic results compared to published DRAM literature such as [21], [41], and [14]. Therefore, published leakage values are used for this analysis rather than simulation results.

Worst-case leakage data for the 1-Gb generation from [14] is used, along with the conservative assumption that leakage current remains constant over time. The values used are i_{Lsub} = 23.4 fA and i_{Lpn} = 9.5 fA per cell for subthreshold and PN junction leakage, respectively; other leakage mechanisms are assumed to be negligible. An operating temperature of 100 ^{o}C is assumed. Based on these values and assumptions, the rate of voltage change on the bitlines is calculated using the fundamental relation

$$\frac{dV_{bl}}{dt} = \frac{i_L}{C_{bl}}, \tag{4.7}$$

where V_{bl} is the bitline voltage, i_L is the total leakage current on the bitline, and C_{bl} is the total bitline capacitance including the open cell. The total leakage current i_L is calculated as

$$i_L = (N_{cbl} + 1)(i_{Lsub} + i_{Lpn}) + 2i_{Lpn}, \tag{4.8}$$

where N_{cbl} is the number of cells per bitline. The first part of the equation considers leakage from all closed cells on the bitline, while the second part considers the open cell. The signal degradation for different wordline lengths is found by multiplying equation 4.7 by the number of cells per wordline and the cycle times for reading or writing a single bit within the memory core as listed in table 4.3.

The worst case amount by which a data signal is degraded by bitline leakage during a write and subsequent read operation is shown in figure 4.7 for different sub-wordline lengths. The data shown is for a fixed bitline length of 128 bits. It is apparent that the signal degradation increases linearly with sub-wordline length. This is a simple consequence of the fact that it takes a fixed amount of time to read and write each cell on a sub-wordline. Increasing the number of cells increases the time for leakage to occur.

Figure 4.8 shows the maximum allowable sub-wordline length if a maximum of a 1% signal degradation is allowed, for different bitline lengths. This non-linear characteristic is related to the effect of changing bitline capacitance on signal magnitude. As the bitline becomes longer and read signals become more diluted, the proposed architecture becomes more susceptible to leakage caused by long read

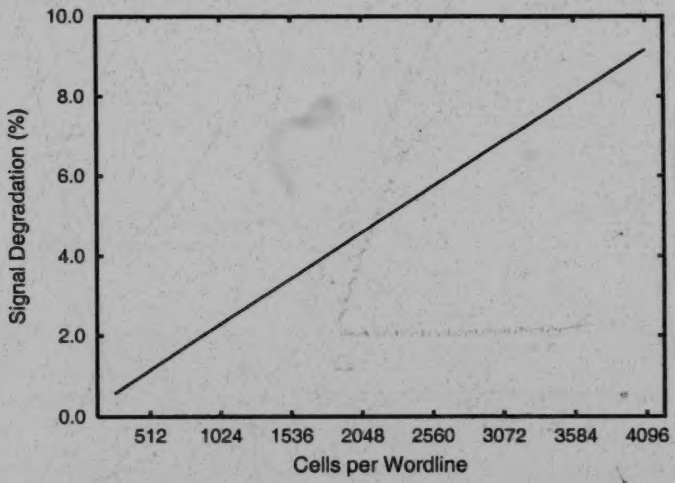


Figure 4.7: Signal Degradation due to Read/Write Leakage for Different Sub-Wordline Lengths

times.

Signal degradation during read is much more significant than that during write. This is for two reasons. First, before the read operation begins, the signal on the bitlines is diluted by charge sharing. Therefore, bitline leakage has a stronger relative effect. Second, the read operation is slower than the write operation, so there is more time for leakage to occur.

4.7.4 Overall Noise Margins

Figure 4.9 shows the worst case overall remaining noise margins for different bitline lengths, after array noise, global process variation, and leakage have been considered. Local process variation is not considered in the figure. To facilitate comparisons with other memory designs that use different supply voltages, the results in figure 4.9 are normalized to $V_{DD}/2$ on the secondary y-axis.

The worst case remaining noise margins are calculated as

$$v_{nm} = v_{signal} - \left(v_{coupling} + v_{gpv} + v_{bl\_leakage} + v_{cell\_leakage}\right), \tag{4.9}$$

where v_{nm} and v_{signal} are the noise margin and signal voltages as defined in chapter 2, $v_{coupling}$ is the coupling noise voltage, v_{gpv} is the noise voltage due to global pro-

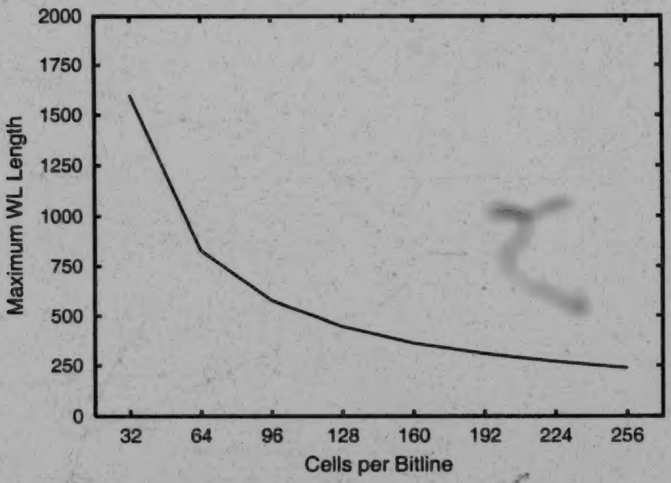


Figure 4.8: Maximum Sub-Wordline Length for 1% Signal Degradation During Read and Write

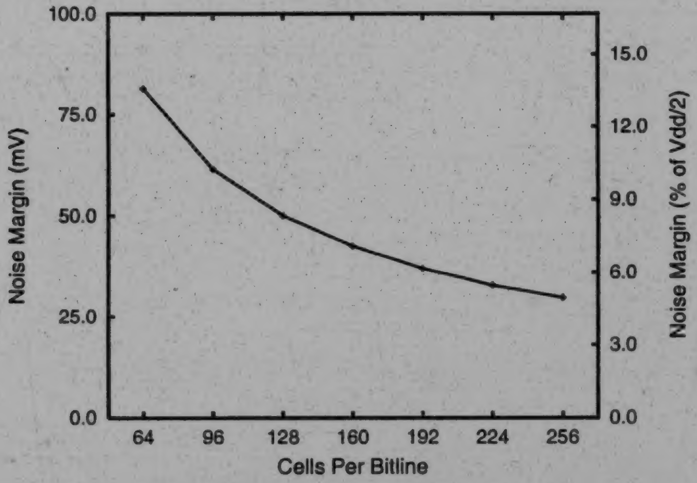


Figure 4.9: Worst Case Noise Margins for Different Bitline Lengths

cess variations, $v_{bl\_leakage}$ is the noise voltage due to leakage from floating bitlines during read and write, and $v_{cell\_leakage}$ is the worst case signal degradation allowed by cell leakage between refreshes. The terms in equation 4.9 actually represent noise at the sense amplifier output, but are expressed here as equivalent voltages (instead of currents) to make them more intuitive.

Figure 4.9, which is based on equation 4.9, uses typical model data for all noise sources. The effects of worst case global process variation are added in as though they are simply another noise source. As expected, increasing the bitline length reduces the available noise margins. The bitline length must therefore be set so that thermal noise and noise due to local bitline variation are less than the remaining signal after all other noise effects have been considered. For the remainder of this section, the term "noise margin" is used to refer to the remaining signal after noise sources have been considered.

The most prevalent noise contributor is array coupling noise, which accounts for over half of the worst case total noise. Global process variations are the second largest contributor. Noise from floating bitline leakage is a very small contributor to the overall noise, as was shown in section 4.7.3. It is important to note that the simulation parameters and data used in generating the final noise results of figure 4.9 are very conservative and worst case in terms of data pattern, process corners, and temperature. Therefore, the data shown can reasonably be assumed to represent the absolute worst possible noise that might be experienced.

Published noise margins for conventional DRAM designs are comparable, but somewhat larger than the simulated results of the proposed architecture. Takahashi et al. have developed a 0.13-μm 1-Gb DRAM design with 25-fF cells, 120-fF bitlines, and an open bitline array with 512 cells per bitline [41], which is very suitable for comparison to the proposed architecture. For a fair comparison, their allowance for local process variation is ignored, as is done with the data in figure 4.9, and their refresh time margin is scaled to the value used in the simulations in this thesis, 10% of the signal. Under these conditions, their DRAM has a noise margin of 10.6% of $V_{DD}/2$, which is comparable to the values in figure 4.9.

Min and Langer report a noise margin of 18.8% of $V_{DD}/2$ for a conventional DRAM with a folded bitline array (with single twisted bitlines) and a feature size of 0.14-μm [22]. However, this measurement only includes bitline coupling noise, neglecting other noise coupling sources and ignoring a refresh time margin.

An interesting result that can be determined, based on the data presented above, is the longest possible bitline length that gives the same noise margins as in a conventional DRAM design. Longer bitlines reduce the area overhead of sense amplifiers, and thus reduce the overall area of the design. Therefore, the longest possible bitline that satisfies noise margins is also the optimal bitline length. To determine this length, an assumption about the extent of local process variation within a subarray unit must be made. In this analysis, it is assumed that local process variation accounts for the same amount of noise as the worst case sense amplifier offset noise assumed in Takahashi's DRAM, which is 42 mV. The longest bitline length in the 0.13-μm implementation of the proposed architecture that will give the same noise margins under worst case conditions as Takahashi's DRAM (4.6% of $V_{DD}/2$ after offset noise is considered) is approximately 80 cells. This result is determined by subtracting 42 mV, or 7.0% of $V_{DD}/2$, from the curve in figure 4.9, and finding the bitline length corresponding to 4.6% of $V_{DD}/2$. Such 80-cell bitlines are much shorter than the 512-cell bitlines in Takahashi's DRAM, and means that 6.4 times more sense amplifiers are required than would be needed with an equivalent length bitline.

As described in appendix B, a cell capacitance of 25 fF is used to obtain the noise results in this section. It is worth noting that 25 fF is generally the lowest cell capacitance specified for gigabit-scale DRAMs [38]. Use of larger cells would result in improved noise margins over those reported here.

4.8 Performance

The maximum operating speed of the proposed architecture, which is what is referred to by "performance," is by design much lower than in conventional DRAM. Good performance is not necessitated by the application domain of the proposed architecture. However, the performance must be sufficient to allow a reasonable refresh rate while still permitting adequate data throughput. If the memory performs too slowly, then there will not be enough time to refresh the memory and read and write data.

Table 4.3 highlights the important performance data obtained through core simulation and analysis. The values shown are for worst case temperature and process

Table 4.3: Core Performance Results

Read Cycle	58.8 ns
Write Cycle	10.0 ns
Read Latency[a]	30.12 μs
Write Latency	10.0 ns
Sub-wordline Write	10.25 μs
Sub-wordline Refresh	40.35 μs

[a] This is the time required to read a sub-wordline.

corners[1], using a conservative (large) estimate of 5 kΩ for the equivalent data bus amplifier load, with a sub-wordline length of 512 cells, and assuming a peripheral clock frequency of 200 MHz. Latencies are determined under the assumption that the memory is not busy when a read or write request is made. Read cycle time has by far the largest impact on overall performance, because a small sense amplifier transistor has to establish a read current on the highly capacitive data bus. Also, the serial nature of internal I/O and refresh operations means that single-bit reads must be repeated many times sequentially in reading an entire sub-row, hindering performance.

In practice, data bus load resistance is determined by the primary sense amplifier circuit used. Simulated read times based on actual primary amplifier circuits are presented in section 4.11. Figure 4.10 gives the general relationship between read time and data bus amplifier input resistance, simulated under worst case conditions with a passive load and a bitline length of 128 cells. The trend in the figure suggests that bus amplifier input resistance should be minimized for optimal performance.

Based on the performance data in table 4.3 and the equations in appendix C, a minimum of 494 sub-wordlines must be read in parallel and the memory must be active at all times in order to achieve a refresh-busy ratio of 100% with a refresh interval of 128 ms in a 1-Gb implementation. This is because each cell is read and

---

[1] The worst case conditions for the performance simulation are at a temperature of 0 °C using fast device models. However, a linear resistance, independent of temperature and process variations, is used for simulation.

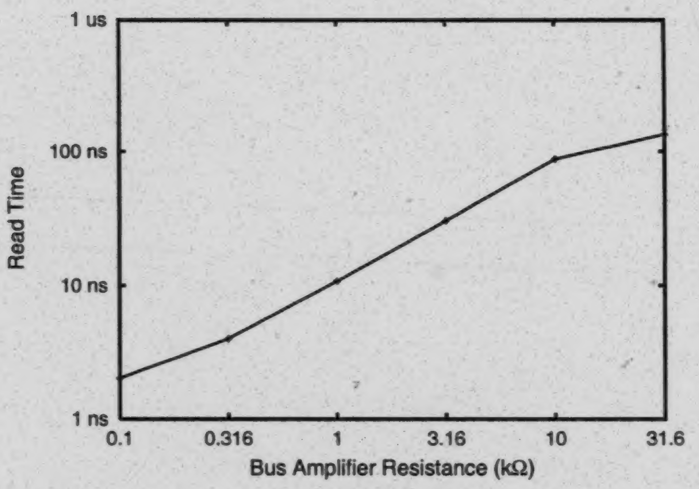


Figure 4.10: Read Time for Different Data Bus Amplifier Input Resistances

written sequentially, and each read operation is relatively slow. A memory with a refresh-busy ratio (defined as the proportion of operating time spent simply refreshing the memory cells) of 100% is not very useful, so a 50% refresh-busy ratio implementation is considered. This implementation requires 987 sub-wordlines to be actively read in parallel, which implies a need for 987 data converter circuits. 505,344 SRAM cells are required to store all of the parallel sub-wordline data during refresh. Internal to the memory there is a large amount of data is available in the SRAM buffer once a read operation is complete, which appears at a rate of 4.2 Gb/s [2]. This throughput can be achieved as long as enough pins and a sufficient output rate are available.

Conventional DRAM designs have much lower read latency than the proposed architecture. Takahashi's 0.13-μm DRAM [41] has an access time of 26.5 ns, which is three orders of magnitude faster than the preceding results, and a throughput of 6.7 Gb/s (in a × 32 configuration), which is approximately the same as the preceding results. A 1-Gb Samsung DRAM [19] in a 0.16-μm technology has a maximum throughput of 4.6 Gb/s, which is also similar. Compared to disk, the read latency of the proposed architecture is far better, being faster by two to three orders of magnitude. The throughput of the proposed architecture is slightly better than a Western Digital serial ATA disk, which has a transfer rate of 1.2 Gb/s [47]. The proposed architecture satisfies its only real performance requirement by having substantially lower latency than disk.

---

[2] The definition of a gigabit used for performance results is 1 Gb = 10^9 bits.

4.9 Power Consumption

The power consumption of the memory core during active operation is determined for the proposed architecture through detailed simulations. Several assumptions are made to obtain a useful value of power consumption that can be compared to the power consumption of a conventional DRAM. First, the power consumption of SRAM cells is estimated from a commercial Samsung SRAM [35], scaled to the 0.13-μm process supply voltage. Second, the data converter power is estimated to be the same as the power of the data bus sense amplifier. While the two circuits would not actually have the same power consumption, it is reasonable to expect that they would be on the same order of magnitude, so this assumption is acceptable for the estimate in this section. Finally, since precharge occurs for such a small fraction of time relative to other operations and does not consume a significant amount of power, the power consumption during precharge is ignored.

Using a sub-wordline length of 256 cells[3], the worst case power consumption of the memory core is estimated to be 87.6 mW during active operation, based on simulation results. Peripheral power is not simulated for the proposed architecture, but it is expected that the peripheral power will be on the same order of magnitude as the core power. The 1-Gb Samsung DRAM mentioned previously reports a total power consumption of 63.4 mW. Therefore, the power consumption of the proposed architecture is relatively large, but not prohibitive, for a low-speed dynamic semiconductor memory.

---

[3] Sub-wordline length has a relatively small impact on power consumption.

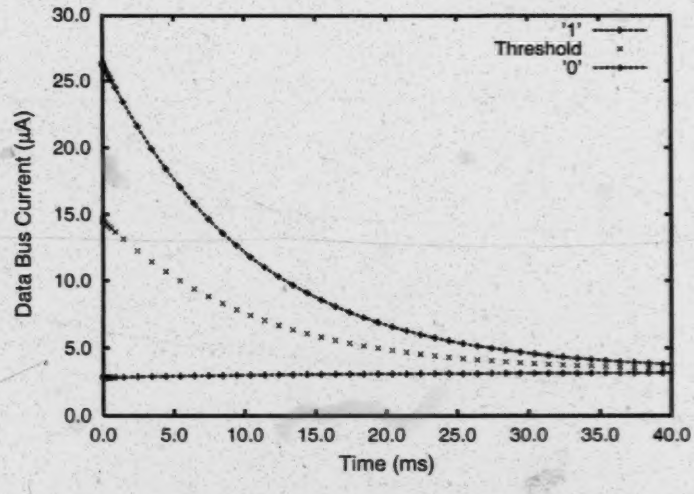


Figure 4.11: Sense Amplifier Current Output Versus Cell Storage Time

4.10 Examination of the Wine Cellar Technique

Although the leakage models available for the 0.13-μm process being considered in this chapter do not agree well with leakage models for a DRAM process, they still represent realistic physical circuit behaviour. Therefore, it is worthwhile to demonstrate the effectiveness of the Wine Cellar Technique using these models.

Figure 4.11 shows the output current of a bitline sense amplifier versus storage time. A bitline length of 128 cells is used, using typical simulation models and a temperature of 25 ^{o}C. The data shown in this figure does not represent a physical current versus time; rather, it shows the simulated current output for a read operation occurring after a given cell leakage time.

The figure shows how a stored '1' value remains above the reference threshold and a stored '0' value remains below the reference threshold well after 7.5 ms, which is the time at which a stored '1' value would be detected as a stored '0' in a conventional reference scheme. In these simulation results, the Wine Cellar Technique significantly extends the data valid time over the conventional scheme, and improves noise margins after shorter amounts of storage time.

A simple exponential decay model is used to further evaluate the Wine Cellar Technique analytically. This model assumes that cell data decays toward 0 V over

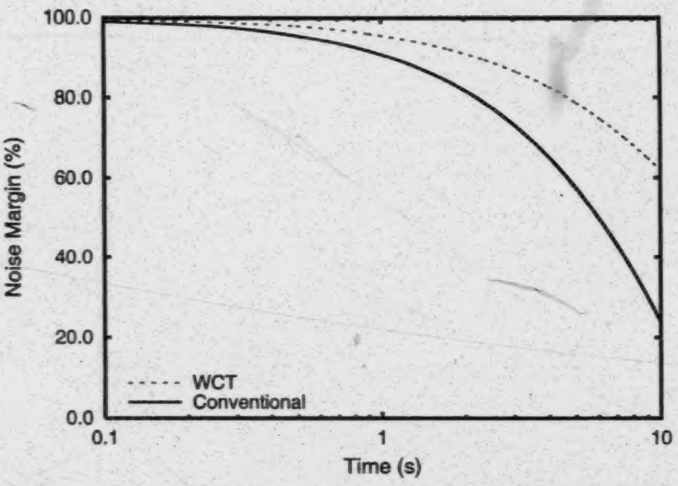


Figure 4.12: Comparison of Conventional and WCT Schemes: Typical Noise Margin

time, and that decay time constants are Normally distributed. A mean and standard deviation for the exponential time constant are estimated from [9] to be $\bar{\tau} = 20.86$ and $\sigma_\tau = 3.01$ for the results that follow, which were generated with the Matlab code in appendix C.

Figure 4.12 shows how the noise margins vary over time in both the conventional scheme and the Wine Cellar Technique (WCT) scheme when reading a stored '1'. The noise margins are shown as a percentage of the noise margins at time 0. The noise margins are clearly better at all times when using the Wine Cellar Technique.

The typical cell voltage characteristic for a stored '1' value is shown in figure 4.13. Also shown are the 10% signal degradation thresholds for the conventional and Wine Cellar reference schemes. These curves are not actual signals, but they indicate where the difference between the stored signal and the reference value is at 90% of its initial value. Therefore, the point at which the stored data curve crosses a threshold curve is the point at which the signal has degraded by 10% from its value at 0 s. Note that while the enlarged view makes the data and Wine Cellar reference curves appear linear, they are in fact exponential. In the figure, the stored '1' data

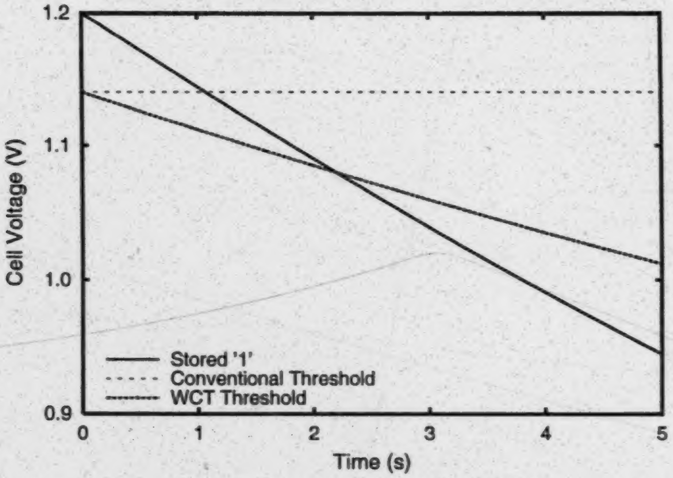


Figure 4.13: Cell Voltage and 10% Thresholds Versus Cell Storage Time

crosses the conventional threshold at 1.07 seconds, compared to 2.20 seconds for the Wine Cellar Technique threshold. This is a 106% increase in valid storage time.

The analytical results presented so far are idealized in that they don't consider local variations in cell leakage. Although the proposed architecture uses local references to minimize the effects of process gradients, it is still important to consider the effects that local process variations can have. Figure 4.14 plots the percentage of read errors occurring versus time for conventional references and for the Wine Cellar Technique, given identical arrays. Again, the Wine Cellar Technique exhibits superior performance.

As a final note, the value of standard deviation used to generate figure 4.14 assumes scatter across an entire chip. Therefore, this data likely represents worse behaviour than might be expected in the proposed architecture with locally stored references. However, the data does not consider tail distribution cells, which have much worse leakage characteristics than predicted by a Normal distribution and are responsible for the high refresh rate requirements of commercial DRAMs (normally 64-256 ms refresh cycles). Though tail cells would also benefit from a combination of locally stored references and the Wine Cellar Technique, redundancy would need to replace some of these cells to ensure error-free operation.

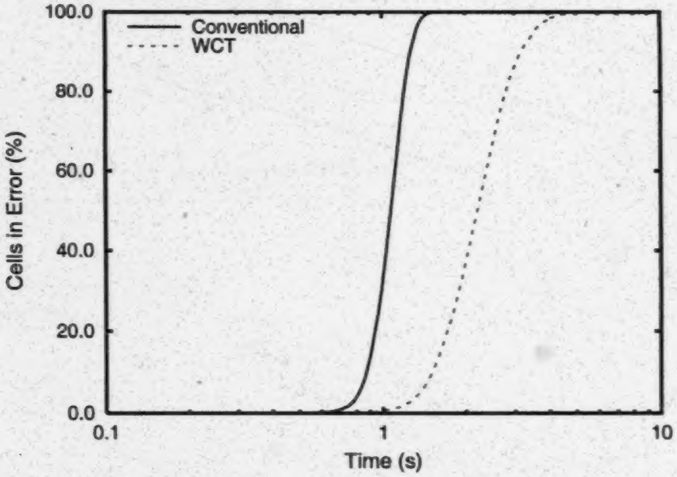


Figure 4.14: Comparison of Conventional and WCT Schemes: Read Errors

4.11 Primary Amplifier Circuit Evaluation

The two data bus sense amplifier circuits proposed in section 3.3.6 are evaluated for their I/O characteristics and their suitability for use in the proposed architecture. Simulations use 5-μm PMOS transistors, with typical models at 25 ^{o}C.

Simulation results for the current mirror amplifier of figure 3.12 are shown in figures 4.15 and 4.16. Figure 4.15 shows how the output current of the amplifier varies with bitline voltage. The characteristic in this figure is very similar to the sense amplifier I/O characteristic shown earlier in figure 4.3. Thus, this amplifier configuration effectively transfers a copy of the data bus current to an isolated output for use in data conversion. Figure 4.16 shows the transient output current when making a full output swing (with a bitline voltage of V_{DD}), beginning with the switching of the column select signal at 10 ns. The output stabilizes very quickly, partly because the bus is pulled to full V_{DD} prior to the read operation.

Simulation results for the bus precharge amplifier of figure 3.13 are shown in figures 4.17 and 4.18. Figure 4.17 shows how the final output voltage of the amplifier varies with bitline voltage for different read times. As suggested by the figure, approximately 40 ns is required for a full output swing even if a bitline voltage of V_{DD} is applied. The transient output current in figure 4.18 is shown for a bitline

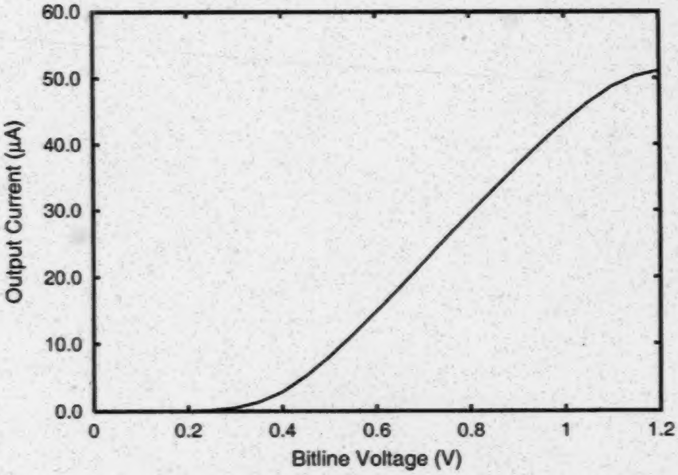


Figure 4.15: Current Mirror Amplifier Output Current Characteristic

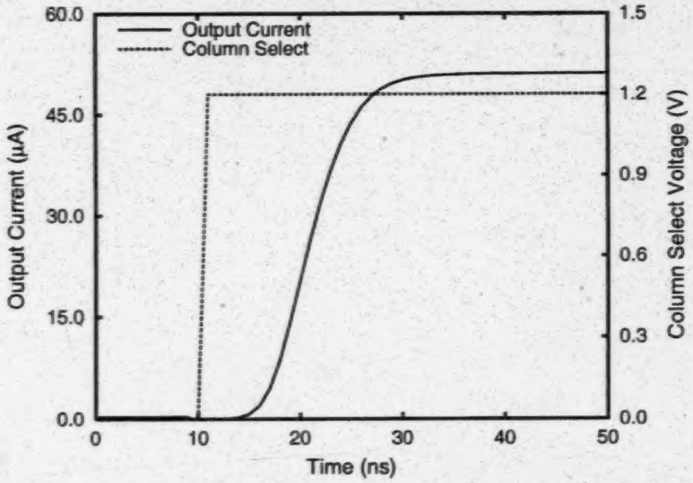


Figure 4.16: Current Mirror Amplifier Transient Characteristic

voltage (V_{bl} in the figure) of 1.2 V and 0.0 V. With a bitline voltage of 1.2 V (V_{DD}), in agreement with figure 4.17, it takes approximately 40 ns for a full output swing to occur.

Both the current mirror and bus precharge amplifier configurations are suitable for data bus reading and amplification based on the results presented. However, the speed of the current mirror amplifier can be controlled (through PMOS transistor sizing) much more than the bus precharge amplifier. Also, with smaller data bus

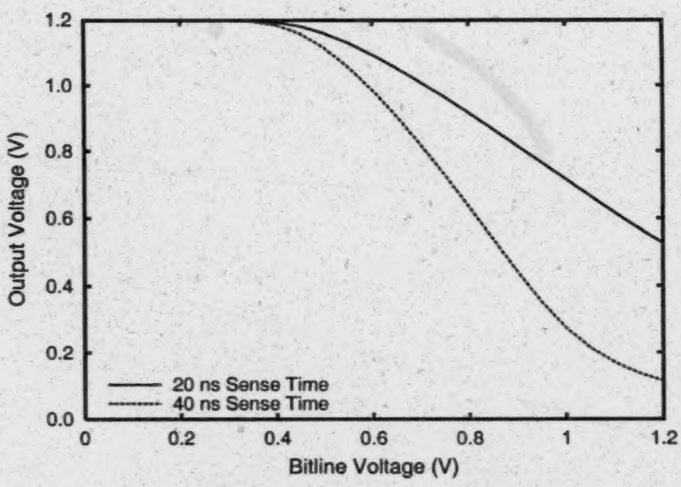


Figure 4.17: Bus Precharge Amplifier Output Voltage Characteristic

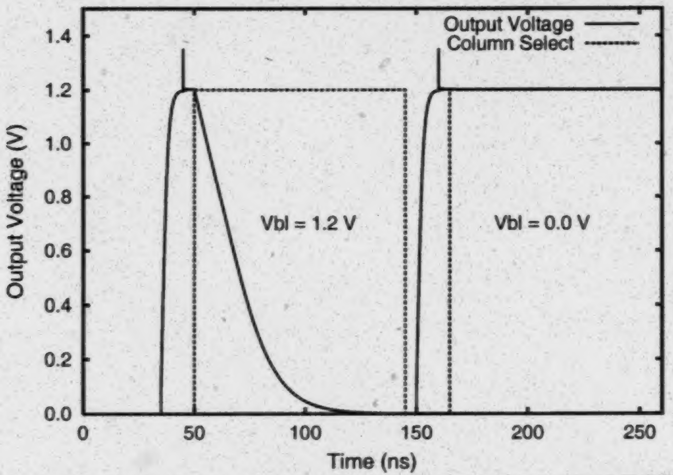


Figure 4.18: Bus Precharge Amplifier Transient Characteristic

voltage swings, the current mirror amplifier will consume less power. Therefore, the current mirror amplifier appears to be preferable to the bus precharge configuration.

4.12 Summary

The analyses and simulations described in this chapter give a great deal of information about the proposed architecture. Most importantly, they verify that the proposed organization, operation and techniques presented in chapter 3 are sound in principle. The single transistor sense amplifier provides a suitable monotonic output current as desired. Given an appropriate bitline bias voltage, a relatively small sense amplifier transistor can generate a current signal on the order of tens of microamps.

Noise simulations give mixed results. Noise due to global process variations is relatively small, a result that is expected given the way that the memory core is designed. Bitline leakage noise specific to the novel read/write scheme is also very small, being well under 10% of the total signal, and is controllable to a reasonable extent by the designer. However, both array coupling noise and noise from local process variations are problematic under worst case conditions. Coupling noise reduces noise margins by approximately 40%, which is quite substantial, though this can be alleviated with an increase in cell capacitance. Potentially more significant is local process variation, which needs to be carefully controlled for the proposed read scheme to function properly. The difficulty of local process variations is discussed further in chapter 6. Overall, if worst case noise from all other sources is considered, then local process variation and thermal effects can introduce as much noise 8.3% of $V_{DD}/2$ with a bitline length of 128 cells, and the memory will still function properly. From another perspective, a bitline length of 80 cells is the longest that will allow proper operation if local process variation is assumed to account for the same amount of noise as the worst case offset in conventional DRAM sense amplifiers. Read/write leakage from floating bitlines suggests a maximum sub-wordline length of approximately 512 cells, so the recommended sub-array size for the 0.13-μm technology considered in this chapter is 512×80. With this sub-array size, the overall chip area is calculated to be $11.72\text{x}10^9 F^2$, which is 14.6% smaller than a conventional DRAM of equal capacity.

Read latency of the proposed architecture is substantially greater than conventional DRAM, but throughput is comparable. Worst case read latency is 30.12 μs, and peak throughput can reach a maximum of 4.2 Gb/s if sufficient high-speed I/O pins are available. Disk throughput is comparable, but when considering that disk latency is on the order of milliseconds, the proposed architecture fits comfortably in the memory hierarchy gap. Power consumption results also show slightly poorer numbers than conventional DRAM, though still on the order of milliwatts. Compared to disk on the order of Watts, however, the power consumption is fairly reasonable [47].

Evaluation of the Wine Cellar Technique shows that even in the presence of local variation in cell leakage, the retention time and/or noise margins for sensing data are larger. The retention time for identical noise margins can be as much as double that of conventional DRAM for main distribution cells.

Simulation of two possible data bus amplifier circuits demonstrates that these circuits can be easily implemented and provide suitable I/O characteristics. The current mirror amplifier has a small advantage in terms of speed and reliability, though either amplifier could be useful in a given implementation.

There are a number of complexities involved in thoroughly characterizing the proposed file memory architecture. While the results presented in this chapter do not exhaustively explore every possible tradeoff, they do quantify the most important tradeoffs, including those involving area, noise margins, and performance. The general trends observed are promising. Further discussion of results, and potential future directions in the development of the proposed architecture are discussed in chapter 6.

Chapter 5

Multilevel Operation

5.1 Overview

The applicability of the proposed memory architecture to multilevel data storage is discussed in earlier chapters. This chapter takes a closer look at some of the possible advantages and challenges of multilevel storage in the context of the architecture and techniques presented earlier.

From a general perspective, very few architectural modifications are required to support multilevel data. This is attributable to the analog nature of both the sense amplifiers and the read, write and restore operations. Array organization, bitline biasing, references, and other aspects of the proposed architecture are fundamentally analog, and therefore do not need to be changed.

Some changes are necessary, however. To store the multiple bits per cell after they have been read, a larger capacity SRAM is needed. The required SRAM size is $log_2 N$ times larger than for two-level storage, where N is the number of levels per cell. Digital-to-analog converters (DACs) must be added to the write path. This adds some area overhead and design complexity, but otherwise has little impact on the architecture. Also, additional reference bitlines are required in each sub-array for the increased number of reference levels. The effect of these changes on area are examined in section 5.2.

Noise margins are affected in a similar way to how they are affected in conventional DRAM. The use of N levels of storage per cell reduces the stored charge by $N-1$, effectively reducing noise margins by the same amount. Multilevel noise margins are considered in section 5.4.

5.2 Area Analysis

When discussing and comparing area results with a multilevel memory, it makes more sense to discuss area per bit. Therefore, for this section, a memory with 2^{30} cells is considered, whose capacity varies with the number of stored levels (such that capacity is equal to $2^{30} \times log_2 N$ [1]). Area per bit is then given by total chip area divided by capacity.

Chip area for the proposed architecture increases with the number of stored levels due to the increased area required for reference bitlines, larger SRAM, and DACs. However, the area per bit decreases significantly as a result of multilevel storage. The total area is calculated in a similar way to the method used in section 4.3, but the value of N_b changes depending on the number of reference bitlines required (one per data level). The number of required SRAM cells increases by $log_2 N$. The number of simple DACs required is constant, and equal to the number of analog-to-digital data converters. There are numerous topologies that can be used for the DACs, each with its own area implications. A possible topology that does not require a significant amount of area is suggested later in this chapter, and for this analysis the DAC area is ignored.

Figure 5.1 shows the area per bit, in terms of bitline half-pitch (F), versus number of stored levels for the proposed architecture with 256 cells per bitline and 512 cells per sub-wordline.

In comparison, the area per bit of a conventional DRAM based on the results of section 4.3 is $12.78F^2$. This shows that, as is expected, multilevel storage in the proposed memory allows a much greater density than two-level storage.

---

[1] It is assumed that every level in every cell represents valid data. If necessary, this data can be converted into binary data by combining multiple cells to obtain a number of levels equal to an integer power of 2.

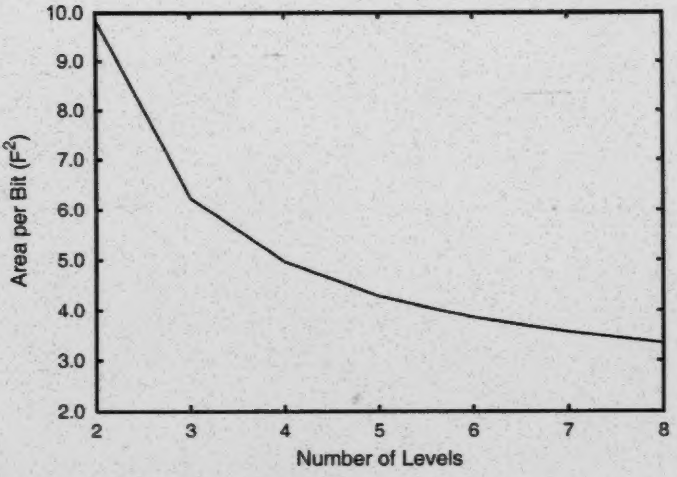


Figure 5.1: Area Per Bit Versus Number of Levels Per Cell

5.3 Functional Simulation

A functional simulation of the proposed architecture in multilevel operation is performed using a similar model and approach to that described in section 4.4. The results of this simulation are shown in figure 5.2. The timing in the figure is relaxed.

The results of functional simulation show that the core of the proposed memory works with multilevel storage as expected, and that no changes to the core operation are required for multilevel to work properly from a functional perspective.

5.4 Noise Margins

As discussed in [1], noise margins are of great importance in multilevel memories. The primary reason for this is that the signal (difference between adjacent levels) in a multilevel memory is substantially reduced. The signal size in a two-level dynamic memory is $V_{DD}/2$. In an N-level memory that signal is reduced by a factor of (N-1) relative to two-level memory, because more levels must be packed into the same voltage range (0 V to V_{DD}). The result is that noise margins are much smaller with multilevel data storage.

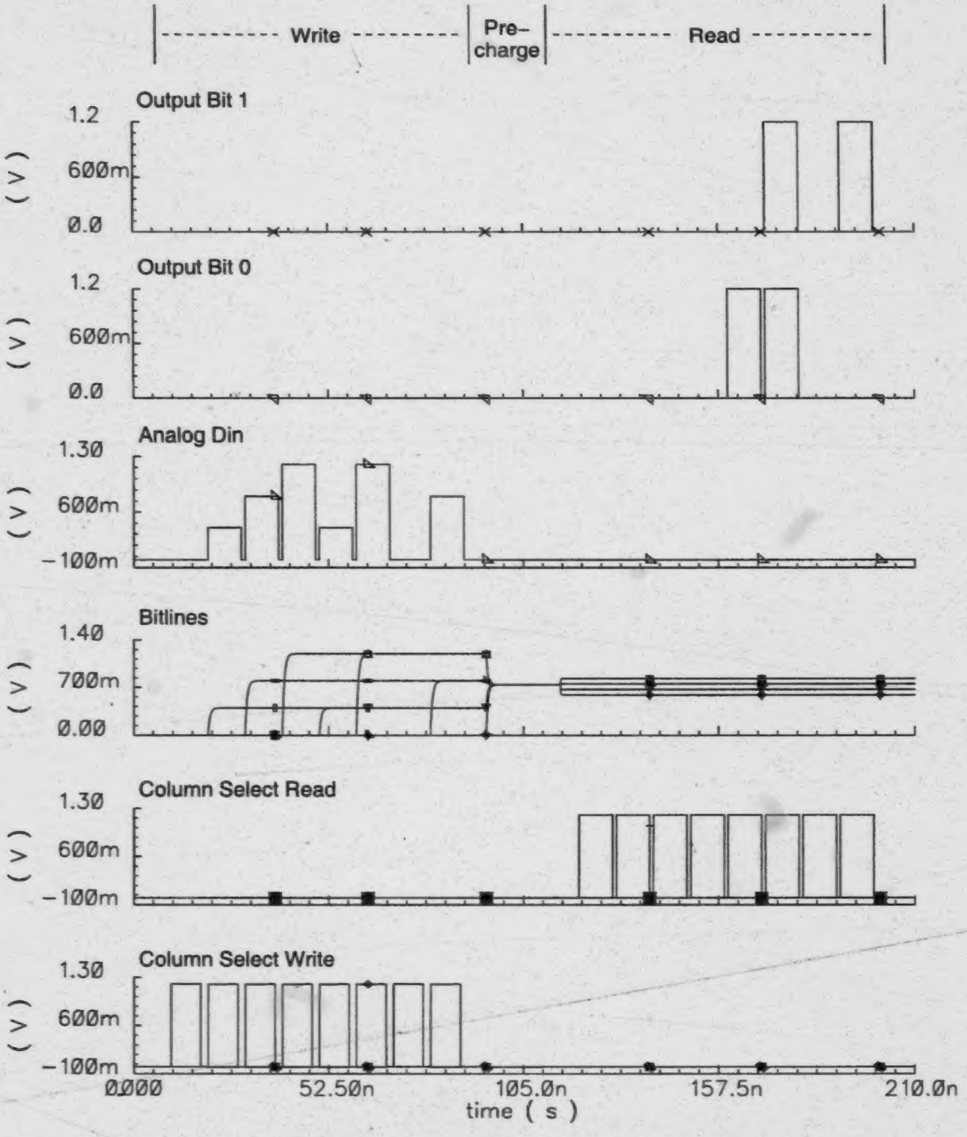


Figure 5.2: Functional Simulation Results for Multilevel Storage

A major noise problem comes from the fact that worst case coupling noise does not scale with the signal when multiple levels are introduced. Average coupling noise is reduced; however, the worst case coupling noise, which occurs when a stored V_{DD} bitline is sandwiched between stored 0 V bitlines, remains constant. The results from figure 4.4 indicate that coupling is a substantial noise contributor in the proposed architecture. This suggests that for robust multilevel operation, either the memory's cell ratio must be improved (for example with larger cell capacitors), or inter-bitline coupling noise must be reduced (for example with a low-k inter-bitline dielectric).

Improving the cell ratio to improve noise margins brings about another challenge in the proposed architecture. In a conventional DRAM, lowering the cell ratio always improves noise margins, since DRAM sense amplifiers can take advantage of the entire bitline voltage range (again, 0 to V_{DD}). The single transistor sense amplifiers employed in the proposed memory can not take advantage of this entire range because they are inactive for bitline voltages below V_{th}. Once their full input range is being used, further increases in cell capacitance (or reductions in bitline capacitance) actually reduce noise margins instead of increasing them, because the bitline voltage goes out of the valid range. The most likely solution to this problem is to use zero-threshold sense amplifier transistors to ensure that the full bitline voltage range can be used. Such solutions are not explored here, but left for future work (as described in section 6.6).

Based on the parameters of the 0.13-μm technology considered in chapter 4, the largest cell ratio that keeps the bitline voltage range within the sense amplifier operating range is approximately 0.85. This equates to about a 48-cell bitline with 25-fF cells. Any further reduction of bitline length decreases noise margins rather than increasing them.

Figure 5.3 shows the worst case signal and noise strengths for a 64-cell bitline with the 25-fF cells used in chapter 4. It is apparent from the figure that the worst case noise, primarily consisting of bitline coupling noise, is too large for successful multilevel operation. Section 5.7 and appendix A.2 discuss a proposed solution to this problem.

5.5 Performance and Power Consumption

By considering the differences between two-level and multilevel embodiments of the proposed architecture, it can be concluded that neither performance nor power

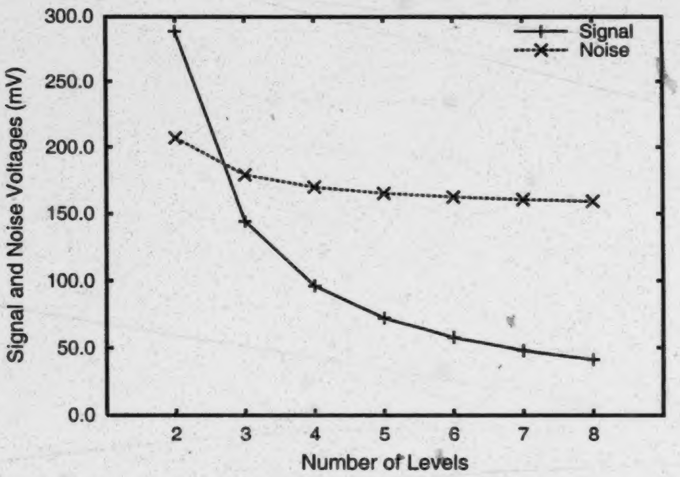


Figure 5.3: Worst Case Signal and Noise Strengths for 64-cell Bitlines and 25-fF Cells

consumption varies substantially. On the critical read path for performance, a small latency penalty must be added for more complex data converters. However, data converter latency is expected to remain small relative to read amplification time for reasonable values of N. Similarly on the write path, the presence of a DAC will slightly increase latency. In this case, the DAC implementation is important for ensuring adequate performance. The DAC need not be excessively complex; a three-bit DAC would be sufficient. It could even be implemented as a simple decoder and analog multiplexer, which would operate quite fast and not require substantial area.

Generally speaking, power consumption decreases in some parts of the architecture, while overall power would remain similar to that of two-level. Assuming a uniform probability of each data level in the multilevel memory, the average power consumption in the array decreases slightly because smaller voltage swings occur. A small increase in peripheral power occurs due to the need for additional SRAM capacity and the inclusion of DACs, as well as multilevel voltage generation circuits and more complex data converters. Overall, power consumption is not affected to a large extent, and therefore the energy per bit will decrease approximately linearly with the number of stored levels.

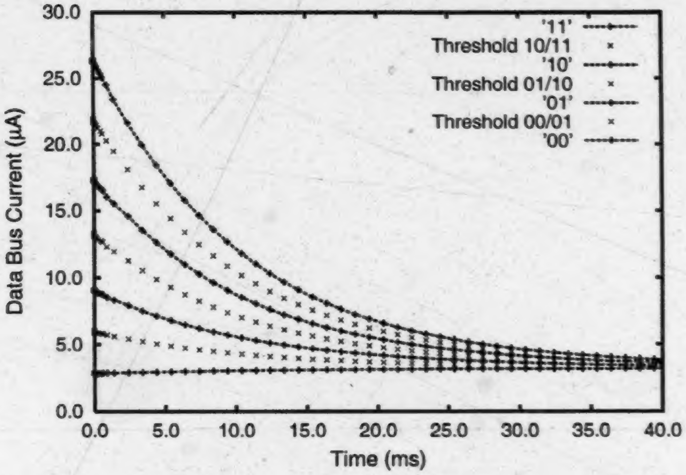


Figure 5.4: Sense Amplifier Output Current Transient Characteristic for Four-Level Storage

5.6 Multilevel Wine Cellar Technique

The analyses described in section 4.10 are repeated for a four-level storage scheme to demonstrate the applicability of the Wine Cellar technique to multilevel storage. Figure 5.4 shows the transient characteristic of stored data and corresponding reference levels from simulation in the UMC's 0.13-μm technology. As in section 4.10, inaccurate simulation models are used, so timing is not to scale. The general trends observed remain valid.

In the simulation, stored data remains valid long after it crosses conventional reference levels. For a stored '11' value, which decreases the most quickly, the data value remains above the reference value long after the conventional threshold is crossed at around 2.5 ms.

Figure 5.5 compares typical noise margins over time for both conventional references and the Wine Cellar Technique using an exponential analytical model as described in section 4.10. As in two-level operation, noise margins remain larger for a longer period of time than with conventional references. The effect is even greater in multilevel storage because of the smaller signal sizes. The typical cell voltage characteristic for a stored '11' value is shown in figure 5.6, along with the

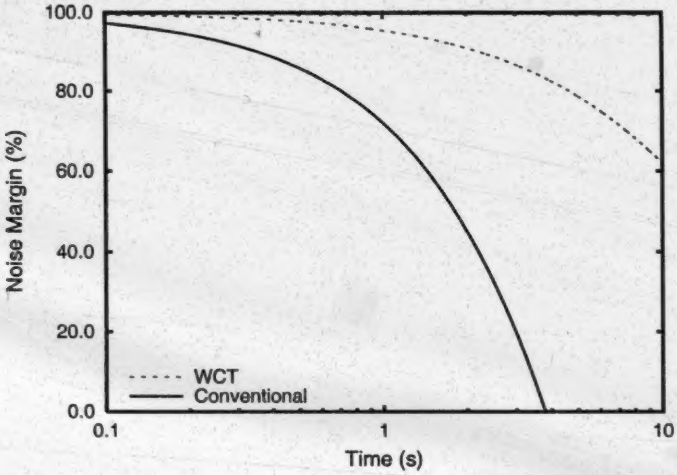


Figure 5.5: Comparison of Conventional and WCT Noise Margins for Four-Level Storage (for a Stored '11')

10% signal degradation thresholds for conventional and Wine Cellar references. The stored '11' value crosses the conventional threshold at 0.35 seconds, compared to 2.20 seconds for the Wine Cellar Technique, which represents a retention time increase of more than six times for a typical cell.

When local process variations are considered, the Wine Cellar Technique still provides larger noise margins than conventional references. Figure 5.7 shows the percentage of cells in error versus time for four-level storage. The results in the figure use the same model described in section 4.10, with $\bar{\tau} = 20.86$ and $\sigma_\tau = 3.01$. Curves are shown for stored data levels representing '11', '10', and '01' [2]. As can be seen in the figure, the Wine Cellar Technique extends valid data time for each storage level, even with locally varying cell retention times.

The results in figure 5.7 use reference cells with a retention time equal to the mean retention time of all cells. Even with cells whose retention times differ from the mean by as much as 3σ, noise margins are still as large as or larger than those with conventional references.

---

[2] The '00' level does not give particularly interesting results in a model that assumes a drift to that value. However, it is worth noting that the noise margin for that value decreases substantially over time with the Wine Cellar Technique but stays constant with conventional references.

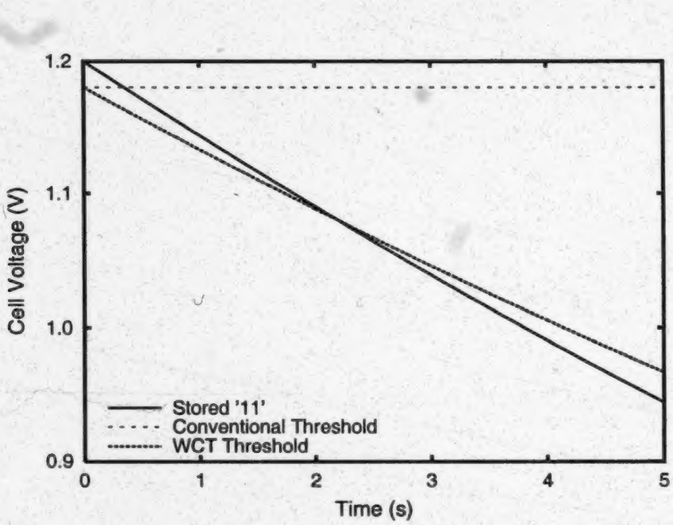


Figure 5.6: Cell Voltage and 10% Thresholds for Four-Level Storage (for a Stored '11')

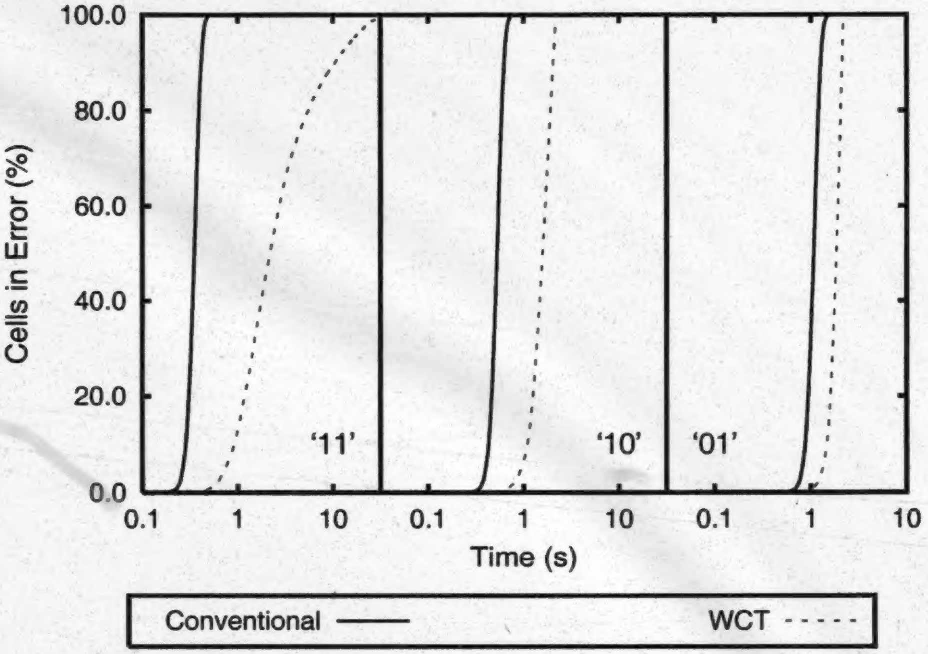


Figure 5.7: Comparison of Conventional and WCT Percentage of Cells in Error for Four-Level Storage

5.7 Summary

Multilevel storage is appealing for any memory architecture due to the potential area improvement. The proposed architecture can easily be adapted for multilevel storage with relatively little overhead, and area improvement is almost linear with the number of data bits.

Noise margins present a much larger challenge in the proposed architecture for multilevel storage than they do for two-level storage. Unlike in a conventional multilevel DRAM, the reduced signal sizes can not be compensated for by continually decreasing the cell ratio in the array. The optimal cell ratio in the UMC's 0.13-μm process for maximum signal strength is approximately 0.85 when using 25-fF cells.

The addition of multilevel storage to the proposed architecture has only a minor effect on performance and power consumption per cell. Given that performance and power consumption are not critical metrics of a file memory implementation, it is sufficient to state that their values should not be expected to change much from two-level to multilevel.

The Wine Cellar Technique works very well with multilevel storage, offering a potential increase in valid retention time by a factor of over six times. A multilevel Wine Cellar Technique may be able to find application in this or other multilevel DRAM architectures.

Bitline coupling noise, the dominant source of noise in the multilevel memory experiments described in this chapter, limits the number of levels that can be reliably read. Since this "noise" is deterministic and is analogous to inter-symbol interference on a communications channel, it can be compensated for. The proposed architecture is particularly well suited for such compensation, as discussed in appendix A.2.

Chapter 6

Conclusion

6.1 Synopsis

This thesis considers the creation of DRAM-based file memory by introducing many divergent ideas and by revisiting a few older ones. From the starting point of a single transistor amplifier, this work presents a feasible architecture and enabling techniques that allow a low-performance but relatively dense semiconductor file memory technology. It also characterizes the most important design tradeoffs that face a designer in implementing the proposed architecture.

6.2 Architecture

The proposed architecture and supporting operational techniques are presented in chapter 3. Much of the architecture borrows from well-established DRAM architectures; however, the core of the architecture, including the sense amplifiers, array organization, and I/O scheme differ significantly. The architecture is designed to facilitate the use of the single transistor sense amplifier while satisfying several requirements, including correct functionality, high density, adequate performance, and easy manufacturing. The primary goal of high density is approached with the use of small sense amplifiers and an open-bitline array organization. The goal of correct functionality is then tackled by introducing novel techniques that are quite different than those used in DRAM. Serial reads and writes are used, as necessitated by the sense amplifiers. Bitline length is reduced to improve noise margins, with area being traded from the sense amplifiers. Noise margins are further improved with a two-pass write scheme. Finally, the unique “Wine Cellar Technique” is developed to improve noise margins and increase refresh times.

A bonus with the proposed architecture is that multilevel memory storage extends very easily from the design. The read and write schemes that are used are not specific to two-level data, so only small changes to the architecture are required for multilevel data. This gives the architecture a great deal of potential.

6.3 Results

Several analyses and simulations are performed using a 0.13-μm process technology in an effort to better understand and characterize the ideas in chapter 3. The results of these analyses and simulations are presented in chapter 4. Density analysis shows a possible area reduction of up to 23.0% when using the same sub-array size as conventional DRAM, or 14.6% when using 80-cell bitlines, either of which would allow a significant reduction in the cost of production DRAM. Simulation of single transistor sense amplifiers (or equivalently, of the 4T configuration) shows a monotonic output current with an effective output range of approximately 50 μA. Memory core analysis and simulation reveals overall noise margins as shown in figure 4.9, with a noise margin of 8.3% of $V_{DD}/2$ with a bitline length of 128 cells in the absence of local process variations. If noise due to local process variations is assumed to be at most equivalent to the worst case sense amplifier offset noise in conventional DRAM, then the optimal sub-array size in the proposed architecture is 512 wordlines × 80 bitlines. Noise margins for 80-cell bitlines in the absence of local process variation are 11.7% of $V_{DD}/2$. The memory has a maximum throughput of 4.2 Gb/s and a read latency of 30.12 μs assuming a 200 MHz peripheral clock, and core power consumption is estimated at 87.6 mW during continuous operation. Application of the Wine Cellar Technique causes effective retention time to approximately double for main distribution cells, even in the presence of local process variation. Finally, both the current mirror and bus precharge data bus amplifier configurations have monotonic output characteristics that do not vary from one read

Table 6.1: Important Characteristics

	Proposed Architecture	Conventional DRAM
Process:	0.13-μm CMOS (0.16-μm $\frac{1}{2}$-pitch)	N/A
Memory Cell:	$6F^2$	$8F^2$ [a]
Area:	$11.72\times10^9F^2$	$13.72\times10^9F^2$ [a]
Sub-Array:	512×80	512×256 [a]
Noise Margin [b] :	11.7% [c]	10.6% [d]
Performance		
Access Time:	30.12 μs	26.5 ns [d]
Throughput:	4.2 Gb/s (max)	6.7 Gb/s ($\times$32) [d]
Power [e]:	87.6 mW [f]	63.4 mW [a]

---

[a] Based on data from [16, 19, 42].
[b] Expressed as a percentage of $V_{DD}/2$.
[c] In the absence of local process variation.
[d] Based on data from [41].
[e] During continuous operation.
[f] Core power only.

to another, making both of these circuits valid choices as data bus read amplifiers.

The important characteristics of the memory are summarized in table 6.1.

6.4 Accomplishments

This work can claim several accomplishments. The idea of using a single transistor transconducting sense amplifier (in a four-transistor configuration) is developed for use with a dynamic cell array. The sense amplifier is characterized and shown to be feasible with major modifications to the dynamic memory core. A suitable core organization is discovered that is compatible with both the single transistor sense amplifier and an open-bitline array organization; as well, the concept of trading some sense amplifier area for array area in order to improve noise margins with shorter bitlines is introduced.

Unique serial read, write and refresh schemes are invented and demonstrated that allow the memory to overcome the functional difficulties of the simple sense amplifier configuration. In particular, a two-pass write is designed to substantially reduce the coupling noise that can plague floating bitlines. The core of the memory is encompassed by a novel architecture that supports these unique schemes through the introduction of data converters and SRAM.

The Wine Cellar Technique is developed and verified as a creative method for improving both noise margins and effective retention time in the memory. Storing references locally as opposed to generating them globally allows very close matching between references and data in the presence of cell leakage, especially after several milliseconds have elapsed.

A new precharge/biasing scheme for bitlines is explained for getting the most out of the analog sense amplifiers while minimizing coupling noise. Also, two data bus sense amplifier circuits are designed and proven as legitimate choices for a full implementation.

The memory can easily support multilevel storage, assuming noise margin constraints can be met. With the proposed architecture, the only changes that need to be made to support multilevel data are the inclusion of digital-to-analog converters, the addition of more reference bitlines, and the addition of extra buffer memory (SRAM) to support the increased amount of data being accessed.

Overall, the proposed dynamic file memory is shown to require considerably less area than conventional DRAM while exhibiting correct functionality, reasonably satisfying noise margin requirements, and providing sufficient performance to fit in the memory hierarchy gap between main memory and disk.

6.5 Challenges

Analysis and simulation results for the proposed semiconductor file memory show that, while conceptually the architecture and techniques work very well, the memory still faces some implementation challenges. The most significant of these is local process variation. Simulation results from section 4.7 show that if opposing worst case process corners are experienced within a single sub-array in the proposed memory, then the resulting noise is unreasonable. Of course, as discussed in chapter 3, it is very unlikely that such a large discrepancy in device parameters would occur in such a small area on chip. Regardless, a better understanding of local process variations in a given process, additional architectural techniques, increased process control, error correction and bad block marking, or a combination thereof is required for the memory to have reasonable yield.

A large part of the area improvement (for 1-bit-per-cell storage) comes from the use of the open bitline array organization, and not from the reduction in sense amplifier size. With continued research into open bitline arrays, and with the evidence presented in chapter 2, it is likely that even some conventional DRAMs will be able to move to open bitline arrays in the near future. This could negate a large part of the area advantage of the proposed architecture. The small, simple sense amplifiers proposed in this thesis may be capable of enabling even smaller open bitline arrays (such as a $4F^2$ cell as presented in [42]), although this possibility remains uncertain.

Reducing average power consumption over long periods of use is another potential challenge for the proposed memory. The power consumption while active is reasonably comparable to that of conventional DRAM, but the power required to refresh the DRAM on a regular basis is much larger due to the serial nature of the refresh operation. The high average power consumption is not a major problem for mains powered applications such as desktop and server computers, but it is quite unrealistic for low-power applications.

The next section describes possible directions for future work that might assist in dealing with these challenges as well as build on the architecture and techniques presented in this thesis.

6.6 Future Work

There are a number of opportunities for future work involving the proposed semiconductor file memory. Such work could involve addressing the challenges presented in the previous section, further characterizing the architecture using different methods, and examining system-level issues related to using the memory.

The most useful pursuit in furthering this research would be to repeat the simulations using actual DRAM process parameters. While this thesis goes to great lengths to ensure that results are compatible with those expected in a DRAM process, simulations performed using DRAM process device models would be much more accurate.

Fabricating and testing a chip using a DRAM process would be even more valuable. Fabricating a test chip in a CMOS logic process with planar storage cells might help verify that some of the architectural concepts in this thesis are valid. However, many of the important qualities of the proposed memory cannot be duplicated in a CMOS process, including realistically-sized memory cells, femtoamp cell leakage, and specialized doping control. Without these qualities, major features of this thesis, such as chip area and noise margins, cannot be accurately tested. Therefore, fabricating a chip would be far more useful if a DRAM process were available.

Local process variation remains an important issue that could be further examined in future work. A study investigating the spatial characteristics of process variation in a DRAM process, especially in the context of the proposed memory, would be invaluable. Presently, to the best of our knowledge, there is no such published data for any DRAM process.

The extendibility of the proposed architecture to support multilevel storage is discussed to some extent in chapter 5. Additional research that evaluates the memory's potential for multilevel storage would still be very beneficial. The evaluation should focus on noise margins and area tradeoffs, as these are the most important considerations for multiple value storage in a file memory. Performance will remain almost unchanged for multilevel, so the performance analysis in this work should be sufficient.

A study of the system-level issues associated with the use of the proposed architecture would be necessary before widespread adoption could take place. Other studies have already demonstrated the general advantages of file memory, but the proposed memory could be specifically studied in a system-level context to determine performance and cost advantages. The economics of the proposed architecture could also be considered for different applications. In particular, an examination of the manufacturing yield would be productive, especially if compared to that of conventional DRAM.

Another area that has been looked at for DRAM but not yet for the proposed architecture is the application of ECC (error-correcting codes) and BBM (bad block marking). The overhead of these techniques would not substantially degrade file memory performance, and they may be able to make the proposed memory more economical. ECC and BBM have the potential to greatly increase yield by dealing with traditional memory faults as well as those that only affect the proposed architecture.

6.7 Summary

The memory presented in this thesis requires less area than conventional DRAM, but has reduced performance. It is unique in many ways, and is realized through the combination of past and present semiconductor memory techniques with new ideas. Both the memory architecture and the new ideas described in this work have strong potential for improving modern computer memory systems and their underlying technology.

Bibliography

[1] G. Birk, D. G. Elliott, and B. F. Cockburn. A comparative simulation study of four multilevel DRAMs. In *Records of the 1999 IEEE International Workshop on Memory Technology, Design and Testing*, pages 102–109, Aug 1999.

[2] G. Birk et al. Multilevel file memories: Challenges and solutions. Technical report, University of Alberta, 2000.

[3] BSIM Research Group, UC Berkeley. *BSIM4.0.0 Release Notes*, 2000.

[4] M. Chang et al. Impact of gate-induced drain leakage on retention time distribution of 256 Mbit DRAM with negative wordline bias. *IEEE Transactions on Electron Devices*, 50(4):1036–1041, Apr 2003.

[5] C. Choi, K. Nam, Z. Yu, and R. W. Dutton. Impact of gate direct tunneling current on circuit performance: A simulation study. *IEEE Transactions on Electron Devices*, 48(12):2823–2829, Dec 2001.

[6] J. P. de Gyvez and D. K. Pradhan. *Integrated Circuit Manufacturability*. IEEE Press, Piscataway, NJ, 1999.

[7] R. H. Dennard. Field-effect transistor memory. United States Patent 3387286, Jun 1968.

[8] D. G. Elliott. personal communications, May 2003.

[9] Takeshi Hamamoto et al. On the retention time distribution of dynamic random access memory (DRAM). *IEEE Transactions on Electron Devices*, 45(6):1300–1309, Jun 1998.

[10] T. P. Haraszti. *CMOS Memory Circuits*. Kluwer Academic Publishers, Boston, MA, 2000.

[11] R. F. Harland. MOS one transistor cell RAM having divided and balanced bit lines, coupled by regenerative flip-flop sense amplifiers, and balanced access circuitry. United States Patent 4045783, Aug 1977.

[12] T. Hasegawa et al. An experimental DRAM with a NAND-structured cell. *IEEE Journal of Solid-State Circuits*, 28(11):1099–1104, Nov 1993.

[13] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, San Francisco, CA, 3rd edition, 2003.

[14] K. Itoh. *VLSI Memory Chip Design*. Springer-Verlag, Berlin, 2001.

[15] C. Joly. Ternary content addressable memory directed redundancy for semiconductor memory yield enhancement. Master's thesis, University of Alberta, 2004.

[16] B. Keeth and R.J. Baker. *DRAM Circuit Design: A Tutorial*. IEEE Press, Piscataway, NJ, 2001.

[17] K. N. Kim et al. Highly manufacturable and high performance SDR/DDR 4 Gb DRAM. In *2001 Symposium on VLSI Technology. Digest of Technical Papers*, pages 7–8, Jun 2001.

[18] J. C. Koob. File memory for extended storage disk caches. Master's thesis, University of Alberta, 2004.

[19] K. Lee et al. A 1 Gbit synchronous dynamic random access memory with an independent subarray-controlled scheme and a hierarchical decoding scheme. *IEEE Journal of Solid-State Circuits*, 33(5):779–786, May 1998.

[20] W. Lee and C. Hu. Modeling CMOS tunneling currents through ultrathin gate oxide due to conduction- and valence-band electron and hole tunneling. *IEEE Transactions on Electron Devices*, 48(7):1366–1373, Jul 2001.

[21] Y. Li et al. Array pass transistor design in trench cell for Gbit DRAM and beyond. In *International Symposium on VLSI Technology, Systems, and Applications, 1999*, pages 251–254, Jun 1999.

[22] D. Min and D. W. Langer. Multiple twisted dataline techniques for multigigabit DRAM's. *IEEE Journal of Solid-State Circuits*, 34(6):856–865, Jun

1970.

[23] S. Mukhopadhyay, A. Raychowdhury, and K. Roy. Accurate estimation of total leakage current in scaled CMOS logic circuits based on compact current modeling. In *Proc. of the 40th Conference on Design Automation*, pages 169–174, Jun 2003.

[24] V. Oklobdzija, editor. *The Computer Engineering Handbook*, chapter 80. CRC Press, 2000.

[25] T. Okuda and T. Murotani. A four-level storage 4-Gb DRAM. *IEEE Journal of Solid-State Circuits*, 32(11):1743–1747, Nov 1997.

[26] C. Papaix and J. M. Daga. A new single ended sense amplifier for low voltage embedded EEPROM non volatile memories. In *Proceedings of the 2002 IEEE International Workshop on Memory Technology, Design, and Testing*, pages 149–153, Jul 2002.

[27] J. M. Rabaey et al. *Digital Integrated Circuits: A Design Perspective*. Prentice Hall, Upper Saddle River, NJ, 2nd edition, 2003.

[28] B. Razavi. *Design of Analog CMOS Integrated Circuits*. McGraw Hill, New York, NY, 2001.

[29] M. Redeker, B. F. Cockburn, and D. G. Elliott. An investigation into crosstalk noise in DRAM structures. In *Proceedings of the 2002 IEEE International Workshop on Memory Technology, Design and Testing*, pages 123–129, Jul 2002.

[30] W. M. Regitz and J. A. Karp. Three-transistor-cell 1024-bit 500-ns MOS RAM. *IEEE Journal of Solid-State Circuits*, SC-5(5):181–186, Oct 1970.

[31] A. Y. Romanenko and W. M. Gosney. A numerical analysis of the storage times of dynamic random-access memory cells incorporating ultrathin dielectrics. *IEEE Transactions on Electron Devices*, 45(1):218–223, Jan 1998.

[32] K. Roy and S. C. Prasad. *Low Power CMOS VLSI Circuit Design*. Wiley Interscience, New York, NY, 2000.

[33] N. Sakashita et al. A 1-GB/s data-rate 1-Gb synchronous DRAM with hierar-

chical square-shaped memory block and distributed bank architecture. *IEEE Journal of Solid-State Circuits*, 31(11):1645–1653, Nov 1996.

[34] Y. Sambonsugi et al. A perfect process compatible 2.49 μm^2 embedded SRAM cell technology for 0.13 μm-generation CMOS logic LSIs. In *1998 Symposium on VLSI Technology. Digest of Technical Papers*, pages 62–63, Jun 1998.

[35] Samsung Electronics Co. Ltd. *Data Sheet: K6R4008V1D 512Kx8 Bit High Speed Static RAM (3.3V Operating)*, 2004.

[36] Semiconductor Industry Association. *International Technology Roadmap for Semiconductors, 2001 Edition*, 2001.

[37] A. K. Sharma. *Semiconductor Memories: Technology, Testing, and Reliability*. IEEE Press, Piscataway, NJ, 1997.

[38] A. K. Sharma. *Advanced Semiconductor Memories*. IEEE Press, Piscataway, NJ, 2003.

[39] K. Stein et al. Storage array and sense / refresh circuit for single-transistor memory cells. *IEEE Journal of Solid-State Circuits*, SC-7(5):336–340, Oct 1972.

[40] T. Sugibayashi et al. A 1-Gb DRAM for file applications. *IEEE Journal of Solid-State Circuits*, 30(11):1277–1280, Nov 1995.

[41] T. Takahashi et al. A multigigabit DRAM technology with $6F^2$ open-bitline cell, distributed overdriven sensing, and stacked-flash fuse. *IEEE Journal of Solid-State Circuits*, 36(11):1721–1727, Nov 2001.

[42] D. Takashima et al. Open/folded bit-line arrangement for ultra-high-density DRAM's. *IEEE Journal of Solid-State Circuits*, 29(4):762–768, Apr 1994.

[43] Y. Tarui and T. Tarui. New DRAM pricing trends: The bi rule. *IEEE Circuits and Devices Magazine*, 7(2):44–45, Mar 1991.

[44] United Microelectronics Corporation. *0.13um Mixed Mode Process Interconnect Capacitance Model (with Metal/Metal Capacitor Module), Rev. 0.1_P0*, 2001.

[45] United Microelectronics Corporation. *0.13um 1.2V/3.3V 1P8M Mixed Mode/*

RFCMOS Technology Process Topological Layout Rule (with Metal/Metal Capacitor Module), Ver. 0.5_P1, 2002.

[46] United Microelectronics Corporation. *0.13um Mixed Mode/RFCMOS Technology 1.2V/3.3V 1P8M Electrical Design Rule (with Metal/Metal Capacitor Module), Ver. 0.2_P1*, 2002.

[47] Western Digital Corporation. *Data Sheet: Western Digital Serial ATA Drive (WD1200JD)*, 2004.

[48] C. Wickman. File store memories. Master's thesis, University of Alberta, 2000.

[49] C. Wickman, B. F. Cockburn, and D. G. Elliott. Cost models for large file memory DRAMs with ECC and bad block marking. In *International Symposium on Defect and Fault Tolerance in VLSI Systems*, pages 319–327, Nov 1999.

[50] H. Yoon et al. A 4Gb DDR SDRAM with gain-controlled pre-sensing and reference bitline calibration schemes in the twisted open bitline architecture. In *IEEE International Solid-State Circuits Conference, Digest of Technical Papers*, pages 378–379, Feb 2001.

Appendix A

Further Discussion

This appendix discusses additional topics that are not presented in the body of this dissertation, but that are nonetheless relevant to the work. These topics include the relationship between density, yield and cost, a coupling noise cancellation scheme that can be applied to read operations, and a comparison of load times for memory pages in the proposed memory and in conventional DRAM.

A.1 Density, Yield and Cost

Successful commercialization of the proposed semiconductor file memory will depend on whether the memory can be produced at sufficiently low cost. This section discusses the relationship between density, yield and cost as it relates to the proposed memory and its production.

For the purposes of this discussion, a simple yield model is used that assumes single defects cause an entire chip to fail, and that defects on a wafer are uniformly distributed and uncorrelated. Furthermore, it is assumed that no redundancy or error-correcting techniques are applied. Quantization of the number of chips per wafer is ignored as well; the model assumes that the number of chips per wafer scales continuously with area.

A Poisson yield model is used, which relates chip yield to the average number of faults per chip as

$$Y = e^{-\lambda}, \tag{A.1}$$

where Y is chip yield and λ is the average number of faults per chip. The number of faults per chip can be expressed as the product of chip area (A) and average defect density (d),

$$\lambda = Ad \,. \tag{A.2}$$

The production cost of each working chip must take into account the cost of producing bad chips that are wasted. This cost is calculated as

$$\text{Cost} = AY^{-1} \,, \tag{A.3}$$

which, by substitution of equation A.2, can be expressed as

$$\text{Cost} = Ae^{Ad} \,. \tag{A.4}$$

In a new manufacturing process, defect density is relatively large. As a manufacturing process matures and its quality improves, defect density becomes relatively much smaller. Therefore, the general relationship between area and cost is better understood by considering the behaviour of equation A.4 for relatively large and relatively small defect densities:

$$\lim_{d \to \infty} (\text{Cost}) \propto e^{A} \,, \tag{A.5}$$

and

$$\lim_{d \to 0} (\text{Cost}) \propto A \,. \tag{A.6}$$

Equation A.5 suggests that for new manufacturing processes, reductions in area result in exponential reductions in cost. For a mature process with high yield, equation A.6 suggests that reductions in area result in linear reductions in cost. This means that, at the very least, the improvement in density offered by the proposed memory translates directly into a reduction in production cost.

A.2 Coupling Noise Cancellation

Section 4.7 shows that capacitive coupling within the memory array is a very large noise contributor. The two-pass write scheme described in chapter 3 eliminates most of the coupling noise during a write operation, but coupling during a read operation still generates a great deal of noise. This section looks at one way that the read noise could be eliminated with a few changes to the proposed architecture.

If analog-to-digital converters (ADCs) with suitable resolution are used as the data converters on the read path, the opportunity to apply digital signal processing to read data becomes available. A coupling matrix could be created, either statically based on estimated coupling noise, or dynamically based on measured coupling noise from the array. Digitized read data could then be multiplied by the inverse of the coupling matrix to effectively cancel out coupling noise, resulting in a much more accurate estimate of stored data from the array.

Equation A.7 shows an example coupling matrix with 20% coupling between adjacent bitlines, and coupling falling off exponentially with distance:

$$C = \begin{bmatrix} 0.0016 & 0.008 & 0.04 & 0.2 & 1 \\ 0.008 & 0.04 & 0.2 & 1 & 0.2 \\ 0.04 & 0.2 & 1 & 0.2 & 0.04 \\ 0.2 & 1 & 0.2 & 0.04 & 0.008 \\ 1 & 0.2 & 0.04 & 0.008 & 0.0016 \end{bmatrix}. \tag{A.7}$$

Equation A.8 shows the inverse matrix of the one in equation A.7. It is apparent that terms fall to zero quickly, so only a few terms in each row need to be considered:

$$C^{-1} = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 & -0.2083 & 1.0417 \\ 0.0000 & 0.0000 & -0.2083 & 1.0833 & -0.2083 \\ 0.0000 & -0.2083 & 1.0833 & -0.2083 & 0.0000 \\ -0.2083 & 1.0833 & -0.2083 & 0.0000 & 0.0000 \\ 1.0417 & -0.2083 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix}. \tag{A.8}$$

The implementation of the coupling matrix and its multiplication would be relatively simple. Most terms can be ignored because the magnitude of coupling from other bitlines within the array drops off so quickly with distance. A simple three-tap finite impulse response (FIR) filter would suffice to multiply all non-zero matrix coefficients in the example matrices shown above.

There are challenges in using a higher-resolution ADC. The biggest challenge is minimizing the area impact of a larger ADC. A very compact design would be required, and even then it is likely that multilevel storage would be necessary to counteract the extra area requirement. Also, the ADC must be able to perform quickly enough to convert each analog data sample during a sequential read operation. If these challenges can be dealt with, the coupling noise cancellation method discussed here could greatly improve the signal-to-noise ratio during read operations, allowing a denser memory core and more levels of multilevel storage.

A.3 Page Load Times

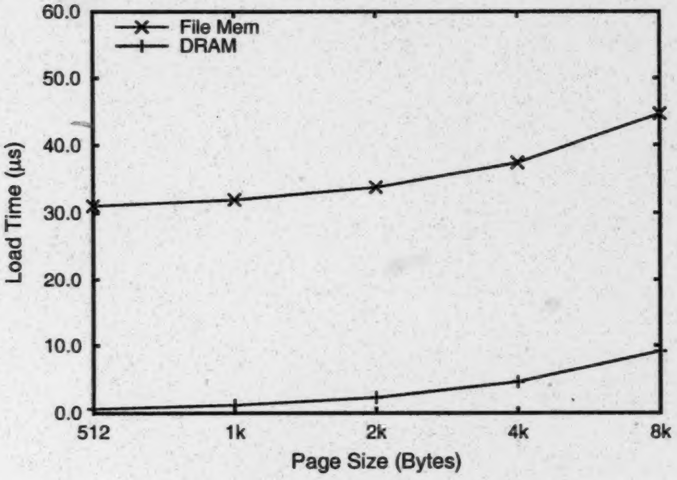
The time to load a page of memory is dependent on the memory access time, the page size, and the data throughput rate, as described by

$$T_{page} = T_{access} + \frac{\text{Page Size}}{\text{Throughput}}, \tag{A.9}$$

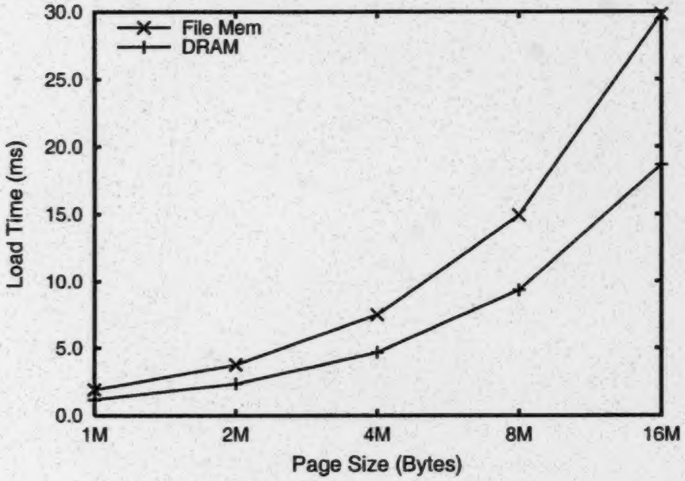
where T_{page} is the time to load a page of memory, and T_{access} is the access latency of the memory.

Figure A.1 shows the relationship between page size and page load time for average and large pages. These figures are based on the performance data in table 6.1. Note that the units of the dependent axis are μs in figure A.1(a) and ms in figure A.1(b).

For a typical 4-kB memory page, the page load time (which is equivalent to the penalty for a page fault) is about eight times greater for the proposed file memory than for conventional DRAM. For large memory pages (such as the 4-MB pages supported by the Intel Pentium architecture or the 16-MB pages supported by the IBM Power 4 architecture), the page load speed of the proposed file memory is more than 50% of the speed of conventional DRAM. This suggests that file memory is most effective with larger page sizes.



(a) Average-Sized Pages



(b) Large Pages

Figure A.1: Page Load Time for Various Page Sizes

Appendix B

Array Model and Simulation Schematics

This appendix contains the most important simulation schematics that were used in generating the simulation results described in chapter 4. These schematics were captured in and plotted from Cadence's Virtuoso Schematic tool. A description of the array model used for simulation is also provided here.

B.1 Array Model

The array model used for analog memory core simulations, shown schematically in figures B.7 and B.8 later in this appendix, uses transistor and parasitic data defined by the UMC's 0.13-μm CMOS process.

Array parameter values are shown in table B.1. These values are derived from UMC's process parameter documentation for an open bitline dynamic array with minimum-pitch aluminum bitlines (0.32-μm pitch) and 1.5x minimum-pitch polysilicon wordlines (0.48-μm pitch). The strict layout requirements for a DRAM array are not actually valid in UMC's CMOS design rules; however, to obtain parameters as close to those of a DRAM array as possible, some design rules were ignored. Where not otherwise noted, array simulations were performed at a temperature of 25 ^{o}C using typical device models.

Wordlines, the precharge line and the CSW line use the boost voltage V_{DDP} to ensure that strong data and precharge levels are written, and that these writes occur

Table B.1: Array Parameters

Parameter Name	Value
V_{DD}	1.2 V
V_{DDP} (Boost Voltage)	1.8 V
Cell Capacitance	25 fF
Access Transistors (Width / Length)	0.16 μm / 0.13 μm
Sense and CSR Transistors (Width / Length)	0.32 μm / 0.26 μm
Wordline Resistance per Cell (Metal-strapped)	183 mΩ
Bitline Resistance per Cell	230 mΩ
Wordline Area / Fringe Capacitance per Cell	21.0 aF [a]
Wordline Coupling Capacitance per Cell	71.0 aF
Bitline Area / Fringe Capacitance per Cell	13.9 aF
Bitline Coupling Capacitance per Cell	80.0 aF [b]

---

[a] 1 aF = 10^{-18} F

[b] This value is actually double the calculated value, to pessimistically account for contact coupling that occurs in a DRAM array but is not normally modeled in a CMOS process.

quickly.

Bitline capacitance measured through array simulations using the parameters in table B.1 agree reasonably well with published DRAM array data. In particular, Takahashi et al. report a bitline capacitance of 120 fF for a 512-cell bitline in a 0.13-μm DRAM process. Simulations with the values from table B.1 yield a bitline capacitance of 95.3 fF for a 256-cell bitline. This is a very reasonable result considering that the values in table B.1 were chosen to be somewhat pessimistic. Min and Langer report a ratio of coupling capacitance to total bitline capacitance of approximately 33% for a bitline pitch of 0.32 μm. Simulation using the parameters in table B.1 give a ratio of 43.0%, suggesting again that these parameters are reasonably accurate but erring on the side of pessimism.

B.2 Simulation Schematics

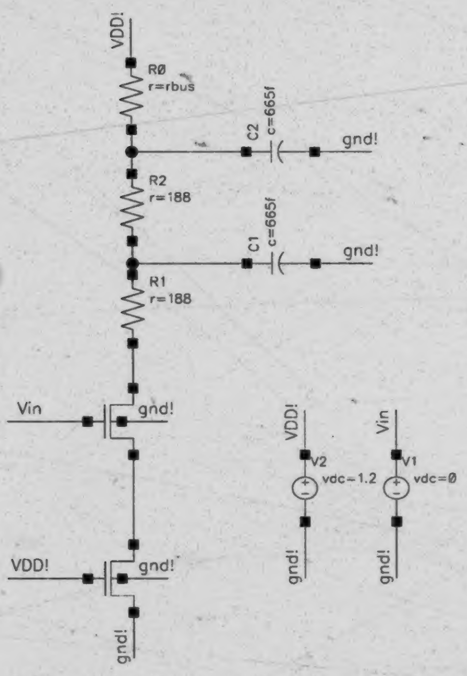

VDD!
R0
r=rbus
C2
c=665f
gnd!
R2
r=188
C1
c=665f
gnd!
R1
r=188
Vin
gnd!
VDD!
gnd!
gnd!
VDD!
V2
vdc=1.2
gnd!
Vin
V1
vdc=0
gnd!

Figure B.1: 1T NMOS CSTA Schematic

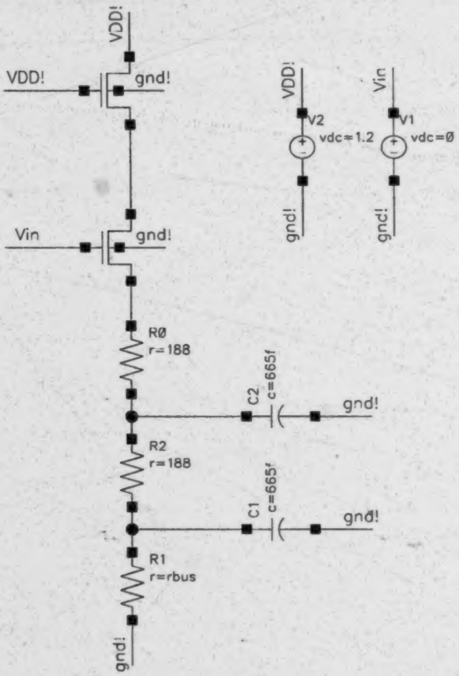

VDD!
VDD!
gnd!
Vin
gnd!
R0
r=188
C2
c=665f
gnd!
R2
r=188
C1
c=665f
gnd!
R1
r=rbus
gnd!
VDD!
V2
vdc=1.2
gnd!
Vin
V1
vdc=0
gnd!

Figure B.2: 1T NMOS CDTA Schematic

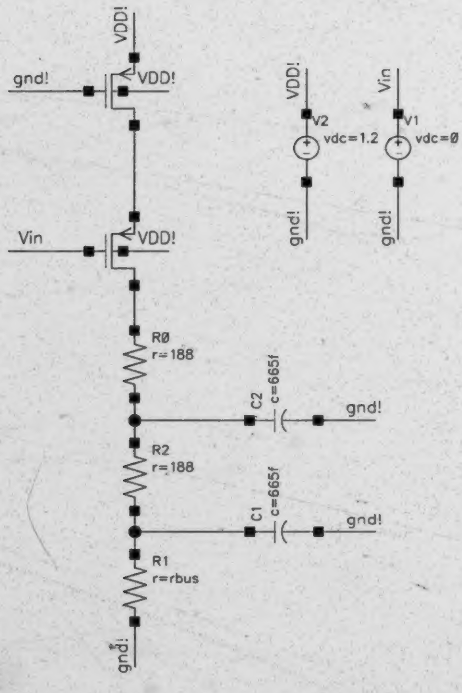

VDD!
gnd!
VDD!
Vin
VDD!
R0
r=188
C2
c=665f
gnd!
R2
r=188
C1
c=665f
gnd!
R1
r=rbus
gnd!
VDD!
V2
vdc=1.2
gnd!
Vin
V1
vdc=0
gnd!

Figure B.3: 1T PMOS CSTA Schematic

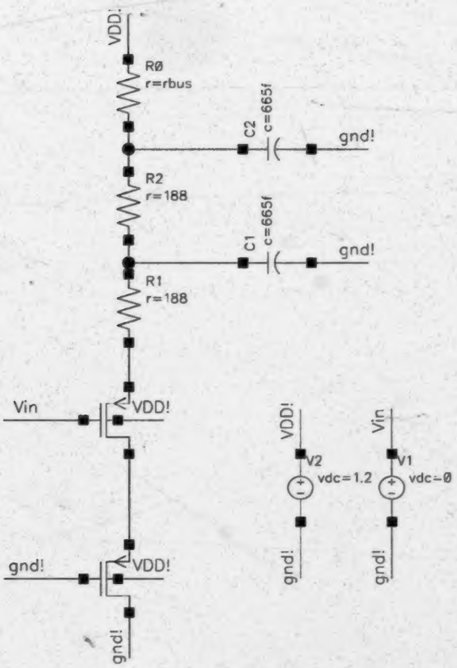

VDD!
R0
r=rbus
C2
c=665f
gnd!
R2
r=188
C1
c=665f
gnd!
R1
r=188
Vin
VDD!
gnd!
VDD!
gnd!
VDD!
V2
vdc=1.2
gnd!
Vin
V1
vdc=0
gnd!

Figure B.4: 1T PMOS CDTA Schematic

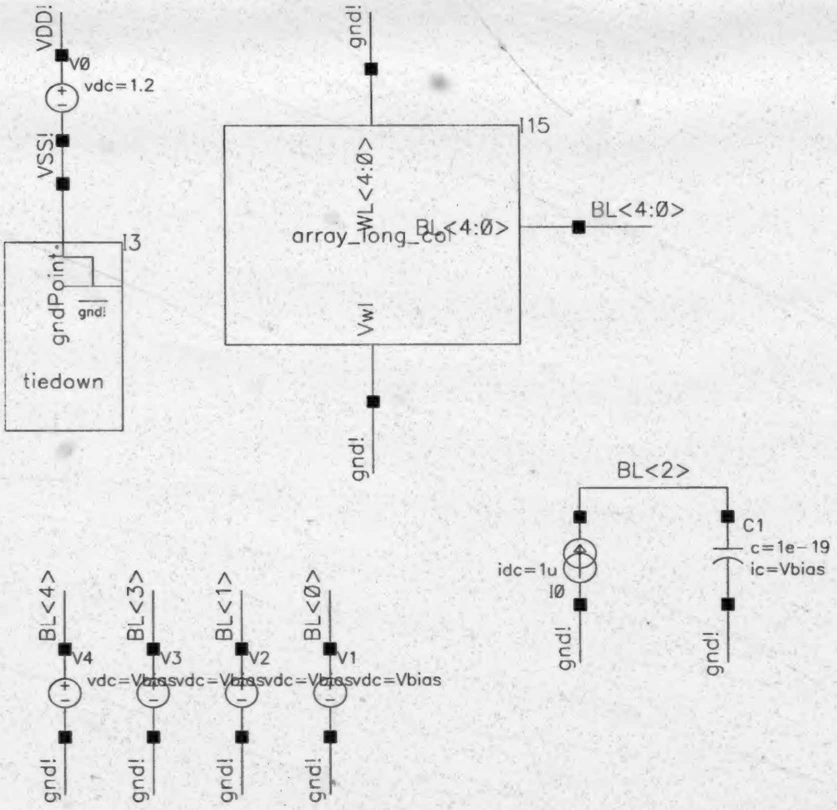

VDD!
V0
vdc=1.2
VSS!
I3
grdPoint
gnd!
tiedown
gnd!
I15
WL<4:0>
array_long_col
BL<4:0>
BL<4:0>
Vwl
gnd!
BL<2>
C1
c=1e-19
ic=Vbias
idc=1u
I0
gnd!
gnd!
BL<4>
BL<3>
BL<1>
BL<0>
V4
V3
V2
V1
vdc=Vbias
vdc=Vbias
vdc=Vbias
vdc=Vbias
gnd!
gnd!
gnd!
gnd!

Figure B.5: Bitline Capacitance Measurement Schematic

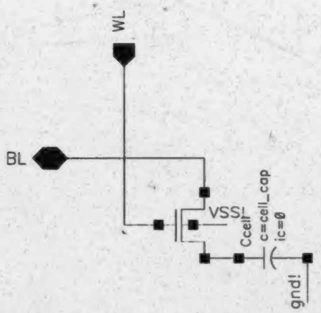

WL
BL
VSS!
Ccell
c=cell_cap
ic=0
gnd!

Figure B.6: Memory Cell Schematic

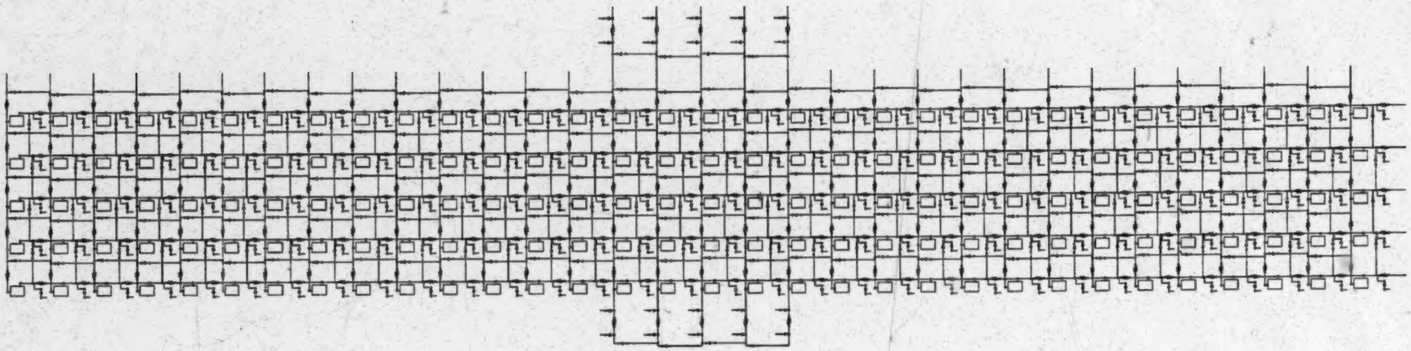


Figure B.7: Memory Array Schematic – 32 x 5

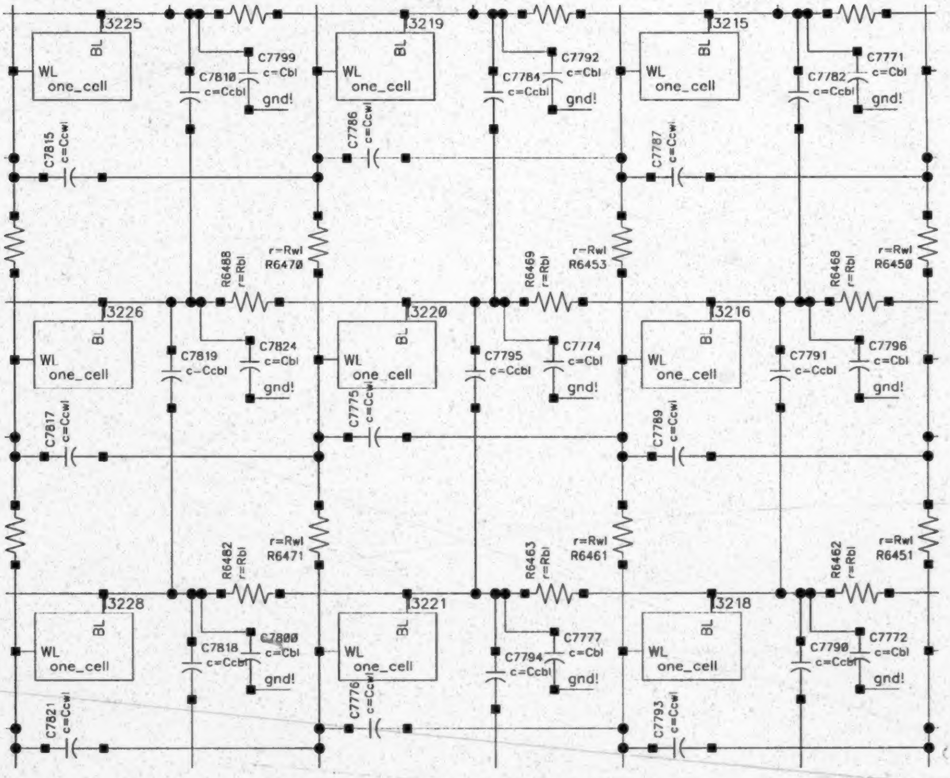

I3225
BL
WL
one_cell
C7799
c=Cbl
C7810
c=Ccbl
gnd!
I3219
C7792
C7784
I3215
C7771
C7782
C7815
c=Ccwl
C7786
C7787
r=Rwl
R6470
R6488
r=Rbl
R6469
R6453
R6468
R6450
I3226
C7824
C7819
c=Ccbl
I3220
C7774
C7795
I3216
C7796
C7791
C7817
C7775
C7789
R6471
R6482
R6463
R6461
R6462
R6451
I3228
C7800
C7818
I3221
C7777
C7794
I3218
C7772
C7790
C7821
C7776
C7793

Figure B.8: Memory Array Schematic – Closeup

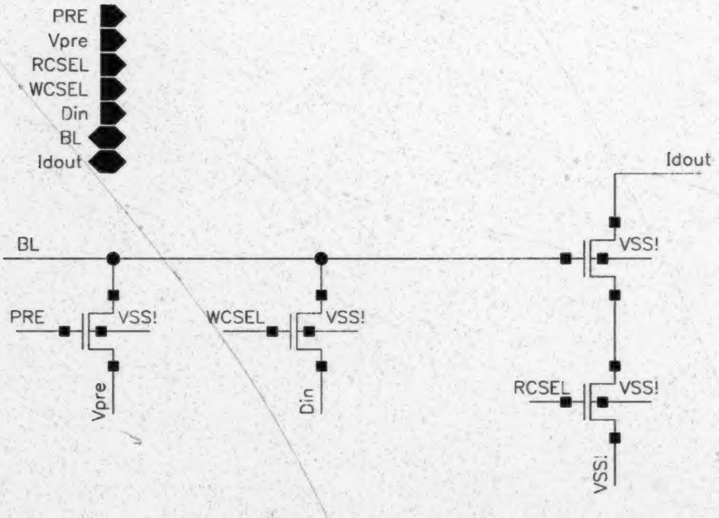

PRE
Vpre
RCSEL
WCSEL
Din
BL
Idout
Idout
BL
VSS!
PRE
VSS!
WCSEL
VSS!
Vpre
Din
RCSEL
VSS!
VSS!

Figure B.9: Sense Amplifier Schematic

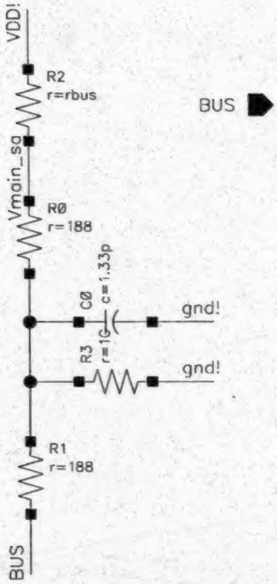

VDD!
R2
r=rbus
BUS
Vmain_sa
R0
r=188
C0
c=1.33p
gnd!
R3
r=1G
gnd!
R1
r=188
BUS

Figure B.10: Data Bus Model Schematic

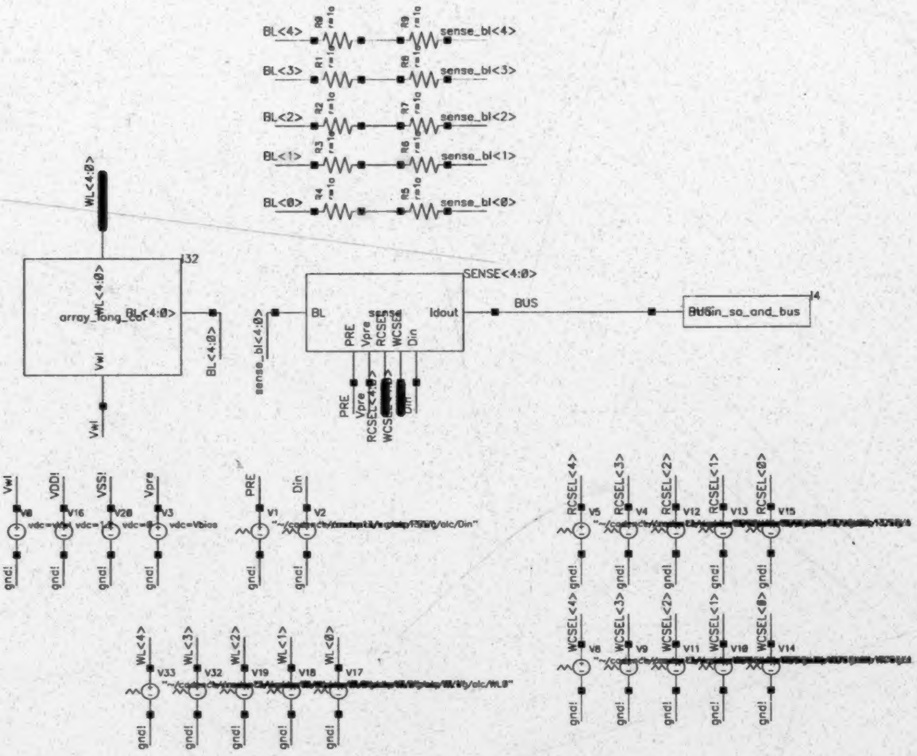

BL<4>
BL<3>
BL<2>
BL<1>
BL<0>
sense_bl<4>
sense_bl<3>
sense_bl<2>
sense_bl<1>
sense_bl<0>
WL<4:0>
I32
WL<4:0>
array_long_car
BL<4:0>
Vwl
BL<4:0>
Vwl
sense_bl<4:0>
SENSE<4:0>
BL
sense
Idout
PRE
Vpre
RCSEL
WCSEL
Din
BUS
I4
BUS
main_sa_and_bus
PRE
Vpre
RCSEL<4:0>
WCSEL<4:0>
Din
Vwl
VDD!
VSS!
Vpre
V0
V16
V20
V3
vdc=Vbias
PRE
Din
V1
V2
gnd!
RCSEL<4>
RCSEL<3>
RCSEL<2>
RCSEL<1>
RCSEL<0>
V5
V4
V12
V13
V15
WCSEL<4>
WCSEL<3>
WCSEL<2>
WCSEL<1>
WCSEL<0>
V6
V9
V11
V10
V14
WL<4>
WL<3>
WL<2>
WL<1>
WL<0>
V33
V32
V19
V18
V17
gnd!

Figure B.11: Core Testbench Schematic

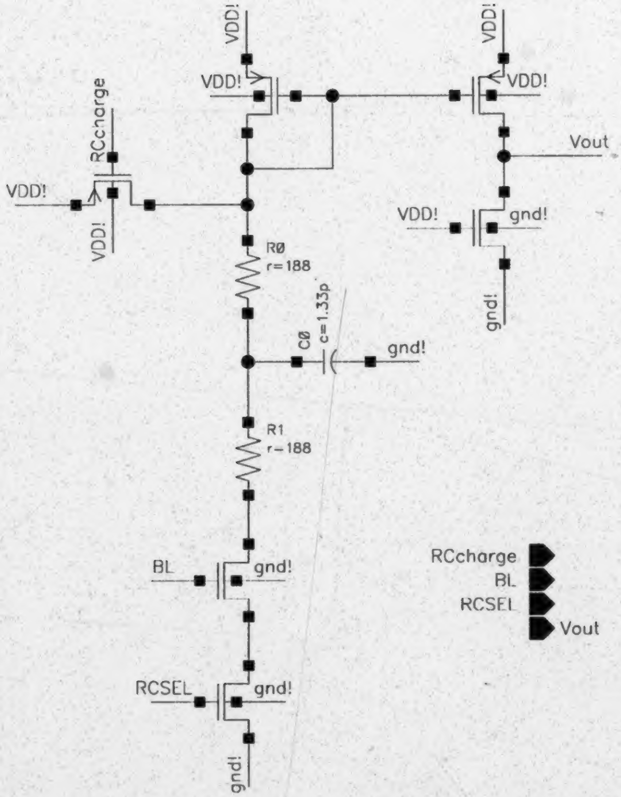

VDD!
VDD!
VDD!
VDD!
RCcharge
VDD!
VDD!
Vout
VDD!
gnd!
gnd!
R0
r=188
C0
c=1.33p
gnd!
R1
r=188
BL
gnd!
RCSEL
gnd!
gnd!
RCcharge
BL
RCSEL
Vout

Figure B.12: Current Mirror Bus Amplifier Test Schematic

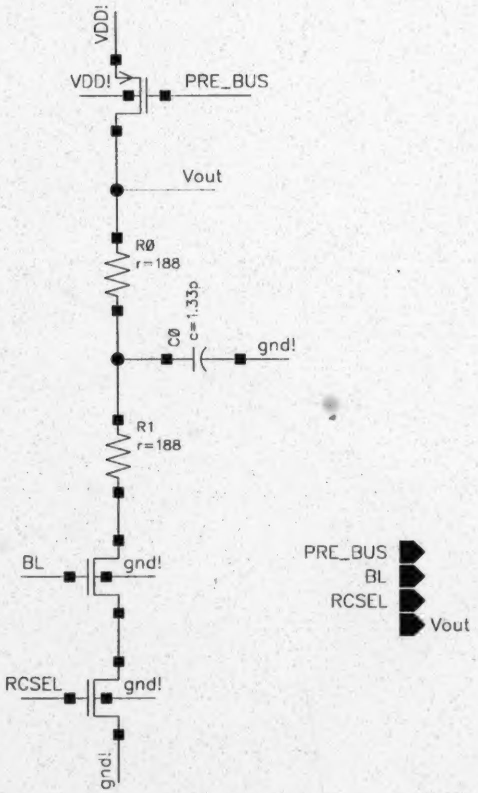

VDD!
VDD!
PRE_BUS
Vout
R0
r=188
C0
c=1.33p
gnd!
R1
r=188
BL
gnd!
RCSEL
gnd!
gnd!
PRE_BUS
BL
RCSEL
Vout

Figure B.13: Bus Precharge Bus Amplifier Test Schematic

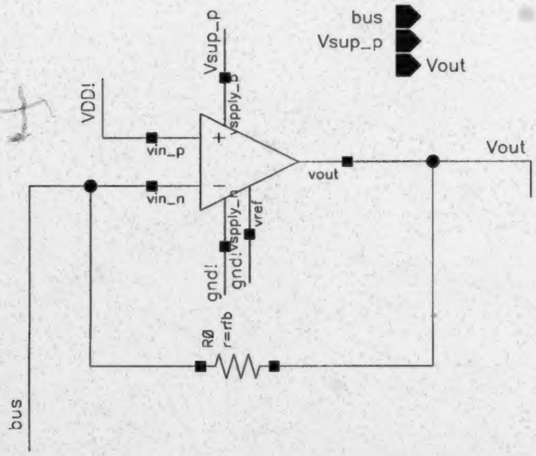

bus
Vsup_p
Vout
Vsup_p
VDD!
vspply_p
vin_p
vin_n
vout
Vout
vspply_n
vref
gnd!
gnd!
R0
r=rb
bus

Figure B.14: Opamp Bus Amplifier Schematic

Appendix C

Details of Analytical Analyses

This appendix contains a description of how performance results were obtained. Also included are tables of calculated data, along with some raw data, that were used to obtain the figures and primary results described in chapter 4. Matlab code used to obtain results for Wine Cellar Technique analysis is provided as well.

C.1 Performance Calculations

Performance calculations are based on a combination of measured read and write times and analytical analysis. The details of the performance calculations are given here. The value T_{clk} in the following calculations refers to the period of the clock that controls peripheral functions.

Read Cycle Once a read operation has been set up, each read cycle requires time for the column address counter to switch and for column decode to occur, which is T_{clk}. Read time is measured through simulation, and referred to as T_{read}. Once the read is established on the data bus, a clock cycle is required to sample the read data value. Total read cycle time is then

$$T_{read} + 2T_{clk}\,. \tag{C.1}$$

Write Cycle Simulated write times are small compared to an estimated peripheral clock period of 5 ns. Write times determined from simulation are less than 2.5 ns; however, the minimum time allowed for synchronous write is T_{clk}, so this value is used in calculating write cycle time. One clock cycle is used for the column address counter to switch and for column decode to occur, and one clock cycle is used for the actual write, for a total write cycle time of

$$2T_{clk}\,. \tag{C.2}$$

Read Latency (Sub-Wordline Read) Two clock cycles are required to latch and decode addresses and to drive a wordline. Once the read is prepared, N_{cwl} read cycles are required, where N_{cwl} is the number of cells per sub-wordline. The total read latency, or time for reading a complete sub-row, is therefore

$$N_{cwl}(T_{read} + 2T_{clk}) + 2T_{clk}\,. \tag{C.3}$$

Write Latency Assuming that the memory is not busy, write latency is only the time required to latch addresses and column decode, and transfer data to the SRAM buffer. This time is

$$2T_{clk}\,. \tag{C.4}$$

Sub-Wordline Write A two-pass write operation requires $2 \times N_{cwl}$ write cycles, plus two clock cycles for address latch and decode and row activation, for a latency of

$$4N_{cwl}T_{clk} + 2T_{clk}\,. \tag{C.5}$$

Sub-Wordline Refresh A sub-wordline refresh operation requires a sub-wordline read followed by a sub-wordline write. The total time is approximately equal to the sum of sub-wordline read and write times, except a few clock cycles for address setup and rewrite row activation are unnecessary. The sub-wordline refresh time is

$$(6N_{cwl} + 1)T_{clk} + N_{cwl}T_{read}\,. \tag{C.6}$$

The number of data converters required to satisfy refresh requirements is calculated as

$$T_{read} \times \frac{capacity}{refresh\_interval}\,, \tag{C.7}$$

where *capacity* is the total memory capacity and $refresh\_interval$ is the time required to refresh the entire chip. This equation is based on the need to be able to read enough sub-wordlines in parallel so as to meet refresh requirements. The number of SRAM cells required is then simply the number of data converters required multiplied by the length of a sub-wordline,

$$N_{data\_converters} \times N_{cwl} \,. \tag{C.8}$$

The maximum internal data throughput of the proposed architecture is calculated as

$$(1-\gamma) \times \frac{capacity}{refresh\_interval}\,, \tag{C.9}$$

where γ is the refresh-busy ratio as defined in section 4.8. This equation reflects the requirement that all data must be read into the SRAM buffer during the course of chip refresh.

C.2 Tables of Calculated and Raw Data

Ap	Wsa	Hlrd	Acsram	Ncsram
2.74E+09	56	123.3	97.7	524288

Afm (x10^9 F^2) Nb (CPW) Nw (CPB)	128	256	384	512	640	768	896	1024
32	18.06	15.53	14.68	14.26	14.01	13.84	13.72	13.63
64	15.25	13.18	12.49	12.15	11.94	11.80	11.70	11.63
96	14.31	12.40	11.76	11.44	11.25	11.12	11.03	10.96
128	13.84	12.01	11.40	11.09	10.91	10.78	10.70	10.63
160	13.56	11.77	11.18	10.88	10.70	10.58	10.50	10.43
192	13.38	11.62	11.03	10.74	10.56	10.45	10.36	10.30
224	13.24	11.51	10.93	10.64	10.46	10.35	10.27	10.20
256	13.14	11.42	10.85	10.56	10.39	10.28	10.19	10.13
288	13.06	11.36	10.79	10.50	10.33	10.22	10.14	10.08
320	13.00	11.30	10.74	10.46	10.29	10.17	10.09	10.03
352	12.95	11.26	10.70	10.42	10.25	10.14	10.06	10.00
384	12.91	11.23	10.67	10.39	10.22	10.11	10.03	9.97
416	12.87	11.20	10.64	10.36	10.19	10.08	10.00	9.94
448	12.84	11.17	10.61	10.34	10.17	10.06	9.98	9.92
480	12.81	11.15	10.59	10.32	10.15	10.04	9.96	9.90
512	12.79	11.13	10.58	10.30	10.13	10.02	9.94	9.88

Ref overhead	128	256	384	512	640	768	896	1024
32	0.16	0.08	0.05	0.04	0.03	0.03	0.02	0.02
64	0.13	0.07	0.04	0.03	0.03	0.02	0.02	0.02
96	0.12	0.06	0.04	0.03	0.02	0.02	0.02	0.02
128	0.12	0.06	0.04	0.03	0.02	0.02	0.02	0.01
160	0.11	0.06	0.04	0.03	0.02	0.02	0.02	0.01
192	0.11	0.06	0.04	0.03	0.02	0.02	0.02	0.01
224	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
256	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
288	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
320	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
352	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
384	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
416	0.11	0.05	0.04	0.03	0.02	0.02	0.02	0.01
448	0.10	0.05	0.03	0.03	0.02	0.02	0.01	0.01
480	0.10	0.05	0.03	0.03	0.02	0.02	0.01	0.01
512	0.10	0.05	0.03	0.03	0.02	0.02	0.01	0.01

Figure C.1: Two Level Area Analysis Data

Listing C.1: Sense Amplifier I/O Characteristic and Gain Data

Vin	Typ (uA)	Slow (uA)	Fast (uA)	F – S (uA)	Typ Gain	Slow Gain	Fast Gain
0.00	0.00	0.00	0.00	0.00			
0.01	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.02	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.03	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.04	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.05	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.06	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.07	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.08	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.09	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.10	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0.11	0.00	0.00	0.00	0.00	0.0	0.0	0.1
0.12	0.00	0.00	0.00	0.00	0.0	0.0	0.1
0.13	0.00	0.00	0.00	0.00	0.0	0.0	0.1
0.14	0.00	0.00	0.00	0.00	0.0	0.0	0.1
0.15	0.00	0.00	0.01	0.01	0.0	0.0	0.2
0.16	0.00	0.00	0.01	0.01	0.0	0.0	0.2
0.17	0.00	0.00	0.01	0.01	0.0	0.0	0.3
0.18	0.00	0.00	0.01	0.01	0.1	0.0	0.4
0.19	0.00	0.00	0.02	0.02	0.1	0.0	0.5
0.20	0.00	0.00	0.02	0.02	0.1	0.0	0.7
0.21	0.00	0.00	0.03	0.03	0.1	0.0	0.9
0.22	0.01	0.00	0.04	0.04	0.2	0.0	1.1
0.23	0.01	0.00	0.05	0.05	0.2	0.0	1.5
0.24	0.01	0.00	0.07	0.07	0.3	0.1	1.9
0.25	0.01	0.00	0.09	0.09	0.4	0.1	2.4
0.26	0.02	0.00	0.12	0.11	0.6	0.1	3.0
0.27	0.03	0.00	0.15	0.15	0.7	0.1	3.8
0.28	0.03	0.01	0.19	0.19	0.9	0.2	4.7
0.29	0.04	0.01	0.25	0.24	1.2	0.2	5.8
0.30	0.06	0.01	0.31	0.30	1.5	0.3	7.1
0.31	0.08	0.01	0.39	0.37	2.0	0.4	8.6
0.32	0.10	0.02	0.48	0.46	2.5	0.5	10.4
0.33	0.13	0.02	0.60	0.57	3.1	0.7	12.4
0.34	0.16	0.03	0.73	0.70	3.9	0.9	14.6
0.35	0.20	0.04	0.89	0.85	4.8	1.1	17.0
0.36	0.26	0.05	1.07	1.02	5.9	1.4	19.6
0.37	0.32	0.07	1.28	1.21	7.2	1.8	22.3
0.38	0.40	0.09	1.52	1.43	8.7	2.3	25.2
0.39	0.50	0.12	1.78	1.67	10.4	2.8	28.0
0.40	0.61	0.15	2.08	1.93	12.3	3.5	30.9
0.41	0.74	0.19	2.40	2.21	14.3	4.4	33.8
0.42	0.89	0.23	2.75	2.52	16.5	5.3	36.7
0.43	1.07	0.29	3.13	2.84	18.9	6.5	39.4
0.44	1.27	0.36	3.54	3.18	21.4	7.8	42.1
0.45	1.50	0.45	3.97	3.53	23.9	9.2	44.7
0.46	1.75	0.55	4.43	3.89	26.4	10.9	47.1
0.47	2.03	0.67	4.92	4.25	28.9	12.7	49.5
0.48	2.33	0.80	5.42	4.62	31.4	14.6	51.6
0.49	2.65	0.96	5.95	4.99	33.9	16.6	53.8
0.50	3.00	1.13	6.50	5.36	36.2	18.7	55.7
0.51	3.38	1.33	7.06	5.73	38.5	20.8	57.5
0.52	3.77	1.55	7.65	6.10	40.7	22.9	59.2
0.53	4.19	1.79	8.25	6.46	42.8	25.1	60.9
0.54	4.63	2.05	8.87	6.82	44.8	27.2	62.4
0.55	5.09	2.33	9.50	7.16	46.6	29.3	63.8
0.56	5.56	2.63	10.14	7.51	48.3	31.2	65.2
0.57	6.05	2.96	10.80	7.84	50.0	33.1	66.5
0.58	6.56	3.30	11.47	8.17	51.6	34.9	67.5
0.59	7.08	3.65	12.15	8.50	53.0	36.7	68.5
0.60	7.62	4.03	12.84	8.81	54.3	38.3	69.5
0.61	8.17	4.42	13.54	9.12	55.6	39.8	71.0
0.62	8.73	4.83	14.26	9.43	56.8	41.3	71.5
0.63	9.31	5.25	14.97	9.72	57.9	42.7	72.0

0.64	9.89	5.68	15.70	10.02	58.8	44.0	73.5
0.65	10.48	6.13	16.44	10.32	60.1	45.2	74.0
0.66	11.09	6.58	17.18	10.60	61.0	46.3	74.5
0.67	11.70	7.05	17.93	10.88	61.5	47.3	75.5
0.68	12.32	7.53	18.69	11.16	62.5	48.3	76.0
0.69	12.95	8.02	19.45	11.43	63.0	49.3	76.0
0.70	13.58	8.51	20.21	11.70	64.0	50.2	77.0
0.71	14.23	9.02	20.99	11.97	65.0	50.9	77.5
0.72	14.88	9.53	21.76	12.23	65.0	51.5	77.5
0.73	15.53	10.05	22.54	12.49	65.5	52.4	78.5
0.74	16.19	10.58	23.33	12.75	66.5	53.0	79.0
0.75	16.86	11.11	24.12	13.01	67.0	53.5	79.0
0.76	17.53	11.65	24.91	13.26	67.0	54.5	79.0
0.77	18.20	12.20	25.70	13.50	67.5	55.0	79.5
0.78	18.88	12.75	26.50	13.75	68.0	55.0	80.0
0.79	19.56	13.30	27.30	14.00	68.5	55.5	80.0
0.80	20.25	13.86	28.10	14.24	69.0	56.5	80.5
0.81	20.94	14.43	28.91	14.48	69.0	56.5	80.5
0.82	21.63	14.99	29.71	14.72	69.0	57.0	80.5
0.83	22.32	15.57	30.52	14.95	69.5	57.5	81.0
0.84	23.02	16.14	31.33	15.19	70.0	57.5	80.5
0.85	23.72	16.72	32.13	15.41	70.0	58.0	80.5
0.86	24.42	17.30	32.94	15.64	70.0	58.0	81.0
0.87	25.12	17.88	33.75	15.87	70.0	58.0	81.0
0.88	25.82	18.46	34.56	16.10	70.5	58.5	81.0
0.89	26.53	19.05	35.37	16.32	70.5	59.0	80.5
0.90	27.23	19.64	36.17	16.53	70.5	59.0	80.5
0.91	27.94	20.23	36.98	16.75	71.0	59.0	80.5
0.92	28.65	20.82	37.78	16.96	70.5	59.0	80.0
0.93	29.35	21.41	38.58	17.17	70.5	59.5	80.0
0.94	30.06	22.01	39.38	17.37	70.5	59.5	79.5
0.95	30.76	22.60	40.17	17.57	70.5	59.5	79.0
0.96	31.47	23.20	40.96	17.76	70.5	59.5	79.0
0.97	32.17	23.79	41.75	17.96	70.5	59.5	78.5
0.98	32.88	24.39	42.53	18.14	70.5	59.5	78.0
0.99	33.58	24.98	43.31	18.33	70.0	59.5	77.5
1.00	34.28	25.58	44.08	18.50	70.0	59.5	76.5
1.01	34.98	26.17	44.84	18.67	69.5	59.0	76.0
1.02	35.67	26.76	45.60	18.84	69.0	59.5	75.5
1.03	36.36	27.36	46.35	18.99	69.0	59.5	74.0
1.04	37.05	27.95	47.08	19.13	69.0	59.0	73.0
1.05	37.74	28.54	47.81	19.27	68.5	59.0	72.0
1.06	38.42	29.13	48.52	19.39	68.0	59.0	70.5
1.07	39.10	29.72	49.22	19.50	68.0	58.5	69.0
1.08	39.78	30.30	49.90	19.60	67.5	58.0	67.0
1.09	40.45	30.88	50.56	19.68	66.5	58.5	65.5
1.10	41.11	31.47	51.21	19.74	66.0	58.5	63.5
1.11	41.77	32.05	51.83	19.78	65.5	57.5	61.0
1.12	42.42	32.62	52.43	19.81	64.5	57.5	59.0
1.13	43.06	33.20	53.01	19.81	64.0	57.5	56.5
1.14	43.70	33.77	53.56	19.79	63.5	56.5	54.0
1.15	44.33	34.33	54.09	19.76	62.0	56.5	51.5
1.16	44.94	34.90	54.59	19.69	61.0	56.5	49.0
1.17	45.55	35.46	55.07	19.61	60.0	55.5	47.0
1.18	46.14	36.01	55.53	19.52	58.5	55.0	44.5
1.19	46.72	36.56	55.96	19.40	57.5	55.0	42.5
1.20	47.29	37.11	56.38	19.27			

Cbl'	Ccbl'	C'dg+C'db	Cg		A (in_min)	B (in_max)	Vdd	Range
13.9	80	221.25	325	<-- Values in aF	0.5	1.15	1.2	0.65

C'bltot (aF)	Csa (aF)		Ccell (fF)
395.15	1742.5		25

N	Cbltot (fF)	BL only | V Cmeas (fF)	Theor. | V Cs/Cb	CR (Cb/Cs)	OBP	Low	High	Range	Invalid?
32	14.4	11.2	1.74	0.58	1.369	0.500	1.262	0.762	INVALID
64	27.0	22.7	0.92	1.08	0.962	0.500	1.077	0.577	valid
96	39.7	34.6	0.63	1.59	0.815	0.500	0.964	0.464	valid
128	52.3	46.7	0.48	2.09	0.739	0.500	0.888	0.388	valid
160	65.0	58.8	0.38	2.60	0.692	0.500	0.833	0.333	valid
192	77.6	70.9	0.32	3.10	0.661	0.500	0.792	0.292	valid
224	90.3	83.1	0.28	3.61	0.638	0.500	0.760	0.260	valid
256	102.9	95.3	0.24	4.12	0.621	0.500	0.735	0.235	valid
288	115.5		0.22	4.62	0.608	0.500	0.713	0.213	valid
320	128.2		0.20	5.13	0.600	0.502	0.698	0.196	valid
352	140.8		0.18	5.63	0.600	0.510	0.690	0.181	valid
384	153.5		0.16	6.14	0.600	0.516	0.684	0.168	valid
416	166.1		0.15	6.64	0.600	0.522	0.678	0.157	valid
448	178.8		0.14	7.15	0.600	0.526	0.674	0.147	valid
480	191.4		0.13	7.66	0.600	0.531	0.669	0.139	valid
512	204.1		0.12	8.16	0.600	0.535	0.665	0.131	valid

Figure C.2: Bitline Capacitance and Optimal Bias Point Analysis Data

Coupling Simulation Results

Meas = Measured
Exp = Expected

Typical Model

		Vhigh (V)		Vlow (V)		Ihigh (uA)		Ilow (uA)	
BL Length	Bias (V)	Meas	Exp	Meas	Exp	Meas	Exp	Meas	Exp
64	0.962	0.9632	1.0730	0.4881	0.4722	30.36	37.82	2.68	2.20
96	0.815	0.8745	0.9653	0.5142	0.4813	24.35	30.51	3.59	2.47
128	0.739	0.8141	0.8905	0.5260	0.4861	20.34	25.44	4.05	2.61
160	0.692	0.7701	0.8355	0.5312	0.4886	17.48	21.76	4.26	2.69
192	0.661	0.7370	0.7942	0.5339	0.4907	15.39	19.05	4.37	2.76
224	0.638	0.7107	0.7612	0.5343	0.4917	13.75	16.92	4.39	2.80
256	0.621	0.6897	0.7349	0.5343	0.4927	12.48	15.25	4.39	2.82

Slow Model

		Vhigh (V)		Vlow (V)		Ihigh (uA)		Ilow (uA)	
BL Length	Bias (V)	Meas	Exp	Meas	Exp	Meas	Exp	Meas	Exp
64	0.962	0.9582	1.0670	0.4847	0.4681	22.09	28.28	1.00	0.78
96	0.815	0.8692	0.9593	0.5125	0.4781	17.10	22.17	1.51	0.91
128	0.739	0.8087	0.8847	0.5254	0.4834	13.81	17.96	1.79	0.99
160	0.692	0.7643	0.8296	0.5312	0.4863	11.48	14.94	1.93	1.03
192	0.661	0.7313	0.7882	0.5341	0.4887	9.81	12.73	2.00	1.07
224	0.638	0.7049	0.7552	0.5347	0.4899	8.52	11.02	2.02	1.09
256	0.621	0.6839	0.7290	0.5346	0.4911	7.54	9.70	2.02	1.11

Fast Model

		Vhigh (V)		Vlow (V)		Ihigh (uA)		Ilow (uA)	
BL Length	Bias (V)	Meas	Exp	Meas	Exp	Meas	Exp	Meas	Exp
64	0.962	0.9614	1.0700	0.4890	0.4732	39.53	48.01	5.84	5.07
96	0.815	0.8731	0.9632	0.5144	0.4821	32.61	39.68	7.19	5.50
128	0.739	0.8131	0.8891	0.5258	0.4866	27.96	33.87	7.83	5.72
160	0.692	0.7694	0.8348	0.5307	0.4890	24.61	29.64	8.12	5.85
192	0.661	0.7369	0.7941	0.5331	0.4911	22.15	26.49	8.25	5.95
224	0.638	0.7109	0.7616	0.5334	0.4920	20.21	24.02	8.27	6.00
256	0.621	0.6902	0.7358	0.5333	0.4930	18.70	22.06	8.26	6.05

Figure C.3: Bitline Coupling Noise Analysis Data (Page 1)

Noise (Voltage)

	Typical			Slow			Fast		
BL Length	% Noise H	% Noise L	Max Noise	% Noise H	% Noise L	Max Noise	% Noise H	% Noise L	Max Noise
64	36.6%	5.3%	36.6%	36.3%	5.5%	36.3%	36.4%	5.3%	36.4%
96	37.5%	13.6%	37.5%	37.4%	14.3%	37.4%	37.5%	13.4%	37.5%
128	37.8%	19.7%	37.8%	37.9%	20.9%	37.9%	37.8%	19.5%	37.8%
160	37.7%	24.6%	37.7%	38.0%	26.2%	38.0%	37.8%	24.1%	37.8%
192	37.7%	28.5%	37.7%	38.0%	30.3%	38.0%	37.8%	27.7%	37.8%
224	37.5%	31.6%	37.5%	37.9%	33.8%	37.9%	37.6%	30.7%	37.6%
256	37.3%	34.4%	37.3%	37.9%	36.6%	37.9%	37.6%	33.2%	37.6%

Noise (Current)

	Typical			Slow			Fast		
BL Length	% Noise H	% Noise L	Max Noise	% Noise H	% Noise L	Max Noise	% Noise H	% Noise L	Max Noise
64	41.9%	2.7%	41.9%	45.0%	1.6%	45.0%	39.5%	3.6%	39.5%
96	43.9%	8.0%	43.9%	47.7%	5.6%	47.7%	41.4%	9.9%	41.4%
128	44.7%	12.6%	44.7%	48.9%	9.4%	48.9%	42.0%	15.0%	42.0%
160	44.9%	16.5%	44.9%	49.7%	12.9%	49.7%	42.3%	19.1%	42.3%
192	44.9%	19.8%	44.9%	50.1%	16.0%	50.1%	42.3%	22.4%	42.3%
224	44.9%	22.5%	44.9%	50.4%	18.7%	50.4%	42.3%	25.2%	42.3%
256	44.6%	25.3%	44.6%	50.3%	21.2%	50.3%	42.0%	27.6%	42.0%

NM Remaining (Voltage)

	Typical			Slow			Fast		
BL Length	NMR H	NMR L	Min	NMR H	NMR L	Min	NMR H	NMR L	Min
64	0.191	0.285	0.191	0.191	0.283	0.191	0.190	0.283	0.190
96	0.151	0.209	0.151	0.151	0.206	0.151	0.150	0.208	0.150
128	0.126	0.162	0.126	0.125	0.159	0.125	0.125	0.162	0.125
160	0.108	0.131	0.108	0.106	0.127	0.106	0.108	0.131	0.108
192	0.095	0.109	0.095	0.093	0.104	0.093	0.094	0.110	0.094
224	0.084	0.092	0.084	0.082	0.088	0.082	0.084	0.093	0.084
256	0.076	0.080	0.076	0.074	0.075	0.074	0.076	0.081	0.076

NM Remaining (Current)

	Typical			Slow			Fast		
BL Length	NMR H	NMR L	Min	NMR H	NMR L	Min	NMR H	NMR L	Min
64	10.350	17.330	10.350	7.560	13.530	7.560	12.990	20.700	12.990
96	7.860	12.900	7.860	5.560	10.030	5.560	10.020	15.400	10.020
128	6.315	9.975	6.315	4.335	7.685	4.335	8.165	11.965	8.165
160	5.255	7.965	5.255	3.495	6.055	3.495	6.865	9.625	6.865
192	4.485	6.535	4.485	2.910	4.900	2.910	5.930	7.970	5.930
224	3.890	5.470	3.890	2.465	4.035	2.465	5.200	6.740	5.200
256	3.445	4.645	3.445	2.135	3.385	2.135	4.645	5.795	4.645

Figure C.4: Bitline Coupling Noise Analysis Data (Page 2)

	Local variation noises		Global Variation Noises						Values are +/- 2.63% ------------------------------				Current noise %:			
BL Length	Noise0	Noise1	gl. noise 0	gl. noise 1	Vn0 typ	Vn1 typ	gl Vn0 typ	gl Vn1 typ	Vn0 typ %	Vn1 typ %	gl Vn0 typ ‘	gl Vn1 typ ‘	loc i0 %	loc i1 %	gl i0 %	gl i1 %
32	7.93	15.90	4.61	4.61	180.55	248.17	124.21	124.21	0.47	0.65	0.33	0.33	0.35	0.69	0.20	0.20
64	10.31	16.86	3.61	3.61	179.76	236.52	71.98	71.98	0.62	0.82	0.25	0.25	0.55	0.90	0.19	0.19
96	11.48	16.86	2.89	2.89	179.73	241.10	51.17	51.17	0.77	1.04	0.22	0.22	0.74	1.09	0.19	0.19
128	12.17	16.70	2.41	2.41	179.39	229.63	40.23	40.23	0.92	1.18	0.21	0.21	0.93	1.27	0.18	0.18
160	12.62	16.52	2.07	2.07	182.81	232.82	33.36	33.36	1.10	1.40	0.20	0.20	1.11	1.45	0.18	0.18
192	12.94	16.36	1.81	1.81	194.90	241.08	28.51	28.51	1.33	1.65	0.20	0.20	1.29	1.64	0.18	0.18
224	13.17	16.22	1.61	1.61	191.21	233.59	24.92	24.92	1.47	1.79	0.19	0.19	1.47	1.82	0.18	0.18
256	13.36	16.12	1.45	1.45	194.93	240.61	22.28	22.28	1.66	2.05	0.19	0.19	1.66	2.00	0.18	0.18
288	13.50	16.01	1.32	1.32	199.93	240.53	20.14	20.14	1.87	2.25	0.19	0.19	1.84	2.18	0.18	0.18
320	13.62	15.93	1.21	1.21	197.01	236.42	18.30	18.30	2.01	2.41	0.19	0.19	2.02	2.36	0.18	0.18
352	13.73	15.85	1.12	1.12	202.08	232.93	16.77	16.77	2.23	2.58	0.19	0.19	2.20	2.54	0.18	0.18
384	13.81	15.79	1.04	1.04	205.60	236.46	15.49	15.49	2.45	2.81	0.18	0.18	2.38	2.72	0.18	0.18
416	13.88	15.73	0.97	0.97	194.87	224.52	14.41	14.41	2.48	2.86	0.18	0.18	2.56	2.90	0.18	0.18
448	13.95	15.68	0.90	0.90	199.61	231.86	13.48	13.48	2.71	3.15	0.18	0.18	2.74	3.08	0.18	0.18
480	14.00	15.63	0.85	0.85	199.40	225.63	12.71	12.71	2.88	3.26	0.18	0.18	2.92	3.26	0.18	0.18
512	14.05	15.59	0.81	0.81	201.67	231.37	12.00	12.00	3.08	3.53	0.18	0.18	3.10	3.44	0.18	0.18

Figure C.5: Process Variation Analysis Data

Read/Write Worst Case Leakage Calculations (@ 85 C) with Worst Case Read Times (@ 0 C)

Cbl'	Ccbl'	C'dg+C'db	Cg	IL'	Ilj'
13.9	80	221.25	325 <-- Values in aF	32.94	9.5

C'bltot (aF)	Csa (aF)	Ccell (fF)	Tclk (ns)
395.15	1742.5	25	5.0

BL Length	Cbl (fF)	Cbl + Cs (fF)	IL (fA)	dVbl/dt	Time for 1mV degradation (us)	Time for 1mV _read signal_ degradation (us)	Time for 10% read signal loss (us)	Time for 10% write signal loss (us)
32	14.4	39.4	1106.02	28.1	35.6	22.6	1356.2	2136.7
64	27.0	52.0	2160.1	41.5	24.1	11.6	694.4	1445.3
96	39.7	64.7	3214.18	49.7	20.1	7.8	466.7	1207.3
128	52.3	77.3	4268.26	55.2	18.1	5.9	351.4	1086.9
160	65.0	90.0	5322.34	59.2	16.9	4.7	281.8	1014.2
192	77.6	102.6	6376.42	62.1	16.1	3.9	235.2	965.5
224	90.3	115.3	7430.5	64.5	15.5	3.4	201.9	930.7
256	102.9	127.9	8484.58	66.3	15.1	2.9	176.8	904.5

Read:

Read Time (ns, +/- 0.5)	Read Cycle Time (ns)	Max WL Length for 10% signal loss	Max WL Length for 1% signal loss	Signal Loss at WL length of 512 (mV)
68.5	78.5	17276	1727	0.3%
69	79	8790	879	0.6%
67	77	6060	606	0.8%
65.4	75.4	4660	466	1.1%
64.9	74.9	3762	376	1.4%
63.6	73.6	3196	319	1.6%
62.8	72.8	2772	277	1.8%
62.8	72.8	2428	242	2.1%

Write:

Write Cycle Time (ns)	Max WL Length for 10% signal loss	Max WL Length for 1% signal loss	Signal Loss at WL length of 512 (mV)
10.0	213670	21367	0.0%
10.0	144526	14452	0.0%
10.0	120734	12073	0.0%
10.0	108693	10869	0.0%
10.0	101421	10142	0.1%
10.0	96553	9655	0.1%
10.0	93067	9306	0.1%
10.0	90447	9044	0.1%

Total:

Max WL Length for 10% signal loss	Max WL Length for 1% signal loss	Total Signal Loss at WL length of 512 (mV)
15984	1598	0.3%
8286	828	0.6%
5771	577	0.9%
4469	446	1.1%
3628	362	1.4%
3093	309	1.7%
2692	269	1.9%
2364	236	2.2%

% Signal Degragdation vs. WL Length for 128 bit BL

WL Length	% Signal Degradation
256	0.6%
512	1.1%
768	1.7%
1024	2.3%
1280	2.9%
1536	3.4%
1792	4.0%
2048	4.6%
2304	5.2%
2560	5.7%
2816	6.3%
3072	6.9%
3328	7.4%
3584	8.0%
3840	8.6%
4096	9.2%

Figure C.6: Read/Write Leakage Analysis Data

Total Noise Margin Analysis Note: non-linearity noise is implicit

BL Length	Coupling Noise	Proc Var Noise	Leakage Noise (WL Length = 512)	Refresh Contrib	Total Noise	NM Remaining (%)	Total Signal (mV)	Total Noise (mV)	NM Remaining (mV)	NM Remaining as % of Vdd/2
32		20.1%	0.3%	10.0%						
64	41.9%	19.3%	0.6%	10.0%	71.8%	28.2%	288	207	81	13.6%
96	43.9%	18.7%	0.9%	10.0%	73.5%	26.5%	232	170	61	10.2%
128	44.7%	18.4%	1.1%	10.0%	74.2%	25.8%	194	144	50	8.3%
160	44.9%	18.2%	1.4%	10.0%	74.5%	25.5%	167	124	43	7.1%
192	44.9%	18.1%	1.7%	10.0%	74.7%	25.3%	146	109	37	6.2%
224	44.9%	18.0%	1.9%	10.0%	74.8%	25.2%	130	97	33	5.5%
256	44.6%	18.0%	2.2%	10.0%	74.7%	25.3%	117	88	30	4.9%

DRAM (from takahashi01multigigabit)

BL Length	Array Noise (mV)	Refresh Contrib (mV)	Amplifier Offset -- Ignored for Comparison	Total Noise (mV)	Total Signal (mV)	NM Remaining (mV)	NM Remaining as % of Signal	NM Remaining as % of Vdd/2
512	34	12		46	120	74	61.7%	12.3%

Figure C.7: Total Noise Margin Analysis Data

Performance Calculations

tclk (ns)	Memory Capacity (Gb)	Ncwl	Target tref (ms)	Area of 1 SRAM Cell (um^2)
5.0	1	512	128	2.5

BL Length	Read Time (ns, +/- 0.5)	Read Cycle (ns)	Read Latency (ns)	Write Cycle (ns)	Write Latency (ns)	Sub-row Read (us)	Sub-row Write (2 pass) (us)	Refresh Cycle w/ Overlap (us)	# of A/D-cycles required	# of SRAM cells req'd (assuming non-pipelined A/Ds
32	48.8	58.8	183.8	10.0	15.0	30.12	10.25	40.35	494	252928
64	48.2	58.2	183.2	10.0	15.0	29.81	10.25	40.04	489	250368
96	47.3	57.3	182.3	10.0	15.0	29.35	10.25	39.58	481	246272
128	46.4	56.4	181.4	10.0	15.0	28.89	10.25	39.12	474	242688
160	45.6	55.6	180.6	10.0	15.0	28.48	10.25	38.71	467	239104
192	44.9	54.9	179.9	10.0	15.0	28.12	10.25	38.35	461	236032
224	44.3	54.3	179.3	10.0	15.0	27.81	10.25	38.05	456	233472
256	43.7	53.7	178.7	10.0	15.0	27.50	10.25	37.74	451	230912

Read times for BL length of 128 for worst case corners

Rbus (ohms)	tread (ns)
100	2.02
316	3.99
1000	10.78
3160	30.37
10000	87.79
31600	134.8

Figure C.8: Performance, Refresh, Data Converter and SRAM Analysis Data

Power Consumption Values in fJ unless otherwise noted

Ewl_cycle	Ncwl -- measured with 256 -- relatively small difference for 512
307	256

BL Length	Ebl_write_avg	Ewbus_and_SA_avg	ESA_read_avg	Erbus_avg	Emain_SA_read	EA/D	Ecsel (r)	Ecsel (w)	Esub-row_read (pJ)	Esub-row_write (pJ)
32	6.40	255.3	969.81	3.68	190.46	190.46	888.2	1998.4	574.11	1155.53
64	9.29	297.8	939.29	2.78	134.47	134.47	888.2	1998.4	537.40	1178.03
96	16.83	303.3	500.76	1.60	87.80	87.80	888.2	1998.4	400.94	1182.78
128	22.49	306.9	402.04	1.24	61.50	61.50	888.2	1998.4	362.11	1186.07
160	27.78	309.4	302.64	1.04	45.34	45.34	888.2	1998.4	328.34	1188.71
192	32.98	311.5	268.07	0.91	35.04	35.04	888.2	1998.4	314.18	1191.11
224	38.13	313.8	244.27	0.70	27.93	27.93	888.2	1998.4	304.39	1193.61
256	43.06	315.3	190.92	0.48	22.99	22.99	888.2	1998.4	288.15	1195.64

Precharge	Ignore -- very small
A/D	Resistor, capacitor, comparator for 2-level -- Assume approx. equal to Emain_SA_read
SRAM	13.87 nW/SRAM_cell from Samsung Sync SRAM at 300 mA

Sub-row Read (us)	Sub-row Write (2 pass) (us)	Read Power (uW)	Write Power (uW)	Average Power per Active Sub-Row (uW)	Active Sub-rows	SRAM Average Power (mW)	Average Core Power (mW)
15.18	5.13	37.83	225.25	85.17	987	3.50	87.56
15.02	5.13	35.77	229.64	85.12	987	3.50	87.51
14.79	5.13	27.10	230.56	79.49	987	3.50	81.96
14.56	5.13	24.86	231.20	78.61	987	3.50	81.09
14.36	5.13	22.87	231.72	77.84	987	3.50	80.33
14.18	5.13	22.16	232.19	77.96	987	3.50	80.44
14.03	5.13	21.70	232.67	78.20	987	3.50	80.68
13.87	5.13	20.77	233.07	78.08	987	3.50	80.57

Figure C.9: Power Analysis Data

BL Length	Bias
128	0.739

Time (ms)	Vcell0	Vcell1	Diluted 0	Diluted 1	I0	I1	Iref
2.00E-07	-0.004	1.195	0.498	0.887	2.935	26.314	14.625
2.01E-07	-0.004	1.196	0.498	0.887	2.935	26.337	14.636
2.02E-07	-0.004	1.198	0.498	0.888	2.935	26.381	14.658
2.08E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.422	14.679
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.420	14.678
2.10E-07	-0.004	1.200	0.498	0.888	2.935	26.418	14.676
2.10E-07	-0.004	1.199	0.498	0.888	2.935	26.406	14.671
2.11E-07	-0.004	1.198	0.498	0.888	2.935	26.381	14.658
2.14E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.20E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.21E-07	-0.004	1.198	0.498	0.888	2.935	26.376	14.656
2.22E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.28E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.30E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.31E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.34E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.40E-07	-0.004	1.198	0.498	0.888	2.935	26.374	14.655
2.40E-07	-0.009	1.196	0.497	0.887	2.863	26.325	14.604
2.40E-07	-0.013	1.194	0.495	0.887	2.829	26.284	14.557
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.40E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.41E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.45E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.50E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.51E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.51E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.51E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.51E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.51E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.52E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.53E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.59E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.80E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.81E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
2.84E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479

Time (ms)	Vcell0	Vcell1	Diluted 0	Diluted 1	I0	I1	Iref
2.95E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
3.41E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
5.23E-07	-0.019	1.190	0.493	0.885	2.766	26.192	14.479
1.25E-06	-0.019	1.190	0.493	0.885	2.767	26.192	14.479
2.71E-06	-0.019	1.190	0.493	0.885	2.767	26.190	14.478
8.53E-06	-0.019	1.189	0.493	0.885	2.769	26.180	14.475
2.02E-05	-0.018	1.189	0.493	0.885	2.773	26.162	14.467
4.35E-05	-0.018	1.187	0.494	0.884	2.780	26.125	14.453
6.68E-05	-0.017	1.185	0.494	0.884	2.786	26.088	14.436
9.01E-05	-0.017	1.184	0.494	0.883	2.792	26.045	14.418
1.83E-04	-0.015	1.176	0.494	0.881	2.809	25.876	14.342
3.23E-04	-0.014	1.164	0.495	0.877	2.824	25.608	14.216
4.63E-04	-0.013	1.152	0.495	0.873	2.832	25.328	14.080
6.03E-04	-0.013	1.140	0.495	0.869	2.838	25.049	13.944
8.82E-04	-0.012	1.115	0.495	0.861	2.846	24.491	13.668
1.44E-03	-0.011	1.068	0.496	0.846	2.860	23.408	13.134
2.44E-03	-0.009	0.987	0.497	0.820	2.883	21.598	12.239
3.44E-03	-0.007	0.913	0.497	0.795	2.904	19.938	11.421
4.44E-03	-0.005	0.845	0.498	0.773	2.923	18.423	10.673
5.44E-03	-0.003	0.781	0.498	0.753	2.942	17.046	9.994
6.44E-03	-0.002	0.723	0.499	0.734	2.959	15.783	9.371
7.44E-03	0.000	0.669	0.499	0.716	2.975	14.643	8.809
8.44E-03	0.001	0.619	0.500	0.700	2.989	13.593	8.291
9.44E-03	0.002	0.573	0.500	0.685	3.003	12.653	7.828
1.04E-02	0.003	0.531	0.500	0.672	3.016	11.794	7.405
1.14E-02	0.004	0.492	0.501	0.659	3.028	11.016	7.022
1.24E-02	0.005	0.455	0.501	0.647	3.040	10.305	6.672
1.34E-02	0.006	0.422	0.501	0.636	3.051	9.666	6.358
1.44E-02	0.007	0.391	0.502	0.626	3.060	9.084	6.072
1.54E-02	0.008	0.363	0.502	0.617	3.070	8.557	5.813
1.64E-02	0.008	0.336	0.502	0.608	3.078	8.078	5.578
1.74E-02	0.009	0.312	0.502	0.600	3.086	7.643	5.365
1.84E-02	0.009	0.289	0.502	0.593	3.093	7.252	5.172
1.94E-02	0.010	0.269	0.503	0.586	3.100	6.895	4.998
2.04E-02	0.011	0.249	0.503	0.580	3.106	6.570	4.838
2.14E-02	0.011	0.232	0.503	0.574	3.112	6.277	4.695
2.24E-02	0.012	0.215	0.503	0.569	3.118	6.009	4.563
2.34E-02	0.012	0.200	0.503	0.564	3.123	5.768	4.445
2.44E-02	0.012	0.186	0.503	0.560	3.128	5.545	4.337
2.54E-02	0.013	0.173	0.503	0.556	3.132	5.347	4.240
2.64E-02	0.013	0.161	0.504	0.552	3.136	5.163	4.150
2.74E-02	0.013	0.150	0.504	0.548	3.140	4.998	4.069
2.84E-02	0.014	0.140	0.504	0.545	3.144	4.847	3.995
2.94E-02	0.014	0.131	0.504	0.542	3.147	4.707	3.927
3.04E-02	0.014	0.122	0.504	0.539	3.150	4.581	3.865
3.14E-02	0.014	0.114	0.504	0.536	3.153	4.467	3.810
3.24E-02	0.015	0.107	0.504	0.534	3.155	4.362	3.759
3.34E-02	0.015	0.100	0.504	0.532	3.158	4.265	3.711
3.44E-02	0.015	0.093	0.504	0.530	3.160	4.176	3.668
3.54E-02	0.015	0.086	0.504	0.528	3.162	4.097	3.629
3.64E-02	0.015	0.082	0.504	0.526	3.164	4.024	3.594
3.74E-02	0.015	0.077	0.504	0.524	3.166	3.956	3.561
3.84E-02	0.016	0.073	0.504	0.523	3.168	3.894	3.531
3.94E-02	0.016	0.068	0.504	0.522	3.169	3.837	3.503
4.04E-02	0.016	0.064	0.504	0.520	3.171	3.784	3.477
4.14E-02	0.016	0.061	0.504	0.519	3.172	3.736	3.454
4.24E-02	0.016	0.057	0.505	0.518	3.173	3.693	3.433
4.34E-02	0.016	0.054	0.505	0.517	3.174	3.653	3.414
4.44E-02	0.016	0.052	0.505	0.516	3.175	3.617	3.396
4.54E-02	0.016	0.049	0.505	0.515	3.176	3.583	3.379
4.64E-02	0.016	0.046	0.505	0.514	3.177	3.551	3.364
4.74E-02	0.016	0.044	0.505	0.514	3.178	3.522	3.350
4.84E-02	0.017	0.042	0.505	0.513	3.179	3.495	3.337
4.94E-02	0.017	0.040	0.505	0.512	3.179	3.471	3.325
5.04E-02	0.017	0.038	0.505	0.512	3.180	3.448	3.314
5.14E-02	0.017	0.037	0.505	0.511	3.181	3.427	3.304
5.24E-02	0.017	0.035	0.505	0.511	3.181	3.407	3.294
5.34E-02	0.017	0.034	0.505	0.510	3.182	3.389	3.285
5.44E-02	0.017	0.033	0.505	0.510	3.182	3.373	3.278
5.54E-02	0.017	0.031	0.505	0.509	3.183	3.358	3.271
5.64E-02	0.017	0.030	0.505	0.509	3.183	3.345	3.264
5.74E-02	0.017	0.029	0.505	0.509	3.184	3.333	3.258
5.84E-02	0.017	0.028	0.505	0.509	3.184	3.321	3.253
5.94E-02	0.017	0.027	0.505	0.508	3.184	3.311	3.247
6.00E-02	0.017	0.027	0.505	0.508	3.185	3.305	3.245

Figure C.10: Wine Cellar Technique Analysis Data

C.3 Matlab Code For Wine Cellar Technique Analysis

Listing C.2: Matlab Code for Noise Margin Analysis

```
Vo = 1.2;
Vf = 0.0;
m_tau = 20.86;
CR = 5;  % Cb/Cs
DR = 1/(1+CR);

% Generate Typical References
t = 0:1e-3:10;
ref0 = (0-Vf).*exp(-t/m_tau) + Vf;
ref1 = (Vo-Vf).*exp(-t/m_tau) + Vf;
mid_ref = (ref0+ref1)/2;

threshold_0 = mid_ref - 0.9*Vo/2;
threshold_1 = mid_ref + 0.9*Vo/2;

nm_1 = (ref1-mid_ref)/(Vo/2)*100;
cnm_1 = (ref1-Vo/2)/(Vo/2)*100;

for i = 1:length(t),
    if (cnm_1(i) < 0)
        cnm_1(i) = 0;
    end
end

nm_percent_better = (nm_1 - cnm_1)./nm_1 .* 100;

tenp = 0.95*Vo*ones(1,length(t));

figure(1);
plot(t,nm_1,t,cnm_1);
figure(2);
plot(t,nm_percent_better);
figure(3);
plot(t,ref0,t,ref1,t,mid_ref,t,threshold_0,t,threshold_1,t,tenp);
axis([0 10 0.8 1.20001]);
```

Listing C.3: Matlab Code for Variation Error Analysis

```
Vo = 1.2;
Vf = 0.0;
m_tau = 20.86;
sd_tau = 3.01;
N_dist = 10000;
cutoff = 0.9;

% Generate Typical References
t = 0:10e-3:10;
ref0 = (0-Vf).*exp(-t/m_tau) + Vf;
ref1 = (Vo-Vf).*exp(-t/m_tau) + Vf;
mid_ref = (ref0+ref1)/2;

threshold_0 = mid_ref - cutoff*Vo/2;
threshold_1 = mid_ref + cutoff*Vo/2;
ct0 = (Vo/2 - cutoff*Vo/2)*ones(1,length(t));
ct1 = (Vo/2 + cutoff*Vo/2)*ones(1,length(t));

err_0 = zeros(1,length(t));
err_1 = zeros(1,length(t));
```

```
cerr_0 = zeros(1,length(t));
cerr_1 = zeros(1,length(t));

% Calculate Gaussian tau values
tau = m_tau + sd_tau*randn(1,N_dist);

% Loop through each dist point for each time point
for i = 1:length(t),
    for n = 1:N_dist;
        stored_0 = (0-Vf).*exp(-t(i)/tau(n)) + Vf;
        stored_1 = (Vo-Vf).*exp(-t(i)/tau(n)) + Vf;
        if (stored_0 > threshold_0(i))
            err_0(i) = err_0(i) + 1;
        end
        if (stored_1 < threshold_1(i))
            err_1(i) = err_1(i) + 1;
        end
        if (stored_0 > ct0(i))
            cerr_0(i) = cerr_0(i) + 1;
        end
        if (stored_1 < ct1(i))
            cerr_1(i) = cerr_1(i) + 1;
        end
    end
end

err_0 = err_0/N_dist*100;
err_1 = err_1/N_dist*100;
cerr_0 = cerr_0/N_dist*100;
cerr_1 = cerr_1/N_dist*100;

%plot(t,ref0,t,ref1,t,mid_ref,t,threshold_0,t,threshold_1);
%plot(t,err_0,t,err_1,t,cerr_0,t,cerr_1);
plot(t,err_1,t,cerr_1);
```

Listing C.4: Matlab Code for Multilevel Noise Margin Analysis

```
V0 = 0.0;
V1 = 0.4;
V2 = 0.8;
V3 = 1.2;
Vo = 1.2;  % Vdd
Vf = 0.0;
m_tau = 20.86;
CR = 5;  % Cb/Cs
DR = 1/(1+CR);

% Generate Typical References
t = 0:1e-3:10;
ref0 = (V0-Vf).*exp(-t/m_tau) + Vf;
ref1 = (V1-Vf).*exp(-t/m_tau) + Vf;
ref2 = (V2-Vf).*exp(-t/m_tau) + Vf;
ref3 = (V3-Vf).*exp(-t/m_tau) + Vf;

mid_ref1 = (ref0+ref1)/2;
mid_ref2 = (ref1+ref2)/2;
mid_ref3 = (ref2+ref3)/2;

threshold_3 = mid_ref3 + 0.9*(1/6)*Vo;

%threshold_0 = mid_ref - 0.9*Vo/2;
%threshold_1 = mid_ref + 0.9*Vo/2;

nm_1 = (ref3-mid_ref3)/(Vo/6)*100;
cnm_1 = (ref3-(5/6)*Vo)/(Vo/6)*100;
```

```
for i = 1:length(t),
    if (cnm_1(i) < 0)
        cnm_1(i) = 0;
    end
end

nm_percent_better = (nm_1 - cnm_1)./nm_1 .* 100;

tenp = (1 -0.1*(1/6))*Vo*ones(1,length(t));

figure(1);
semilogx(t,nm_1,t,cnm_1);
figure(2);
plot(t,nm_percent_better);
figure(3);
plot(t,ref3,t,threshold_3,t,tenp);
axis([0 5 0.9 1.20001]);
```

Listing C.5: Matlab Code for Multilevel Variation Error Analysis

```
V0 = 0.0;
V1 = 0.4;
V2 = 0.8;
V3 = 1.2;
Vo = 1.2;  % Vdd
Vf = 0.0;
m_tau = 20.86;
sd_tau = 3.01;
N_dist = 10000;
cutoff = 0.9;

ref_tau = m_tau; % + 3*sd_tau;

% Generate Typical References
t = 0:10e-3:100;
ref0 = (V0-Vf).*exp(-t/ref_tau) + Vf;
ref1 = (V1-Vf).*exp(-t/ref_tau) + Vf;
ref2 = (V2-Vf).*exp(-t/ref_tau) + Vf;
ref3 = (V3-Vf).*exp(-t/ref_tau) + Vf;

mid_ref1 = (ref0+ref1)/2;
mid_ref2 = (ref1+ref2)/2;
mid_ref3 = (ref2+ref3)/2;
mrt = (ref0+ref3)/2;

threshold_3m = mid_ref3 + cutoff*(1/6)*Vo;
threshold_2p = mid_ref3 - cutoff*(1/6)*Vo;
threshold_2m = mid_ref2 + cutoff*(1/6)*Vo;
threshold_1p = mid_ref2 - cutoff*(1/6)*Vo;
threshold_1m = mid_ref1 + cutoff*(1/6)*Vo;
threshold_0p = mid_ref1 - cutoff*(1/6)*Vo;

ct3m = ((5/6)*Vo + cutoff*(1/6)*Vo)*ones(1,length(t));
ct2p = ((5/6)*Vo - cutoff*(1/6)*Vo)*ones(1,length(t));
ct2m = ((3/6)*Vo + cutoff*(1/6)*Vo)*ones(1,length(t));
ct1p = ((3/6)*Vo - cutoff*(1/6)*Vo)*ones(1,length(t));
ct1m = ((1/6)*Vo + cutoff*(1/6)*Vo)*ones(1,length(t));
ct0p = ((1/6)*Vo - cutoff*(1/6)*Vo)*ones(1,length(t));

err_3 = zeros(1,length(t));
err_2 = zeros(1,length(t));
err_1 = zeros(1,length(t));
err_0 = zeros(1,length(t));

cerr_3 = zeros(1,length(t));
cerr_2 = zeros(1,length(t));
```

```
cerr_1 = zeros(1,length(t));
cerr_0 = zeros(1,length(t));

% Calculate Gaussian tau values
tau = m_tau + sd_tau*randn(1,N_dist);

% Loop through each dist point for each time point
for i = 1:length(t),
    for n = 1:N_dist;
        stored_3 = (V3-Vf).*exp(-t(i)/tau(n)) + Vf;
        stored_2 = (V2-Vf).*exp(-t(i)/tau(n)) + Vf;
        stored_1 = (V1-Vf).*exp(-t(i)/tau(n)) + Vf;
        stored_0 = (V0-Vf).*exp(-t(i)/tau(n)) + Vf;

        if (stored_3 < threshold_3m(i))
            err_3(i) = err_3(i) + 1;
        end
        if ((stored_2 > threshold_2p(i)) || (stored_2 < threshold_2m(i)))
            err_2(i) = err_2(i) + 1;
        end
        if ((stored_1 > threshold_1p(i)) || (stored_1 < threshold_1m(i)))
            err_1(i) = err_1(i) + 1;
        end
        if (stored_0 > threshold_0p(i))
            err_0(i) = err_0(i) + 1;
        end

        if (stored_3 < ct3m(i))
            cerr_3(i) = cerr_3(i) + 1;
        end
        if ((stored_2 > ct2p(i)) || (stored_2 < ct2m(i)))
            cerr_2(i) = cerr_2(i) + 1;
        end
        if ((stored_1 > ct1p(i)) || (stored_1 < ct1m(i)))
            cerr_1(i) = cerr_1(i) + 1;
        end
        if (stored_0 > ct0p(i))
            cerr_0(i) = cerr_0(i) + 1;
        end
    end
end

err_0 = err_0/N_dist*100;
err_1 = err_1/N_dist*100;
err_2 = err_2/N_dist*100;
err_3 = err_3/N_dist*100;

cerr_0 = cerr_0/N_dist*100;
cerr_1 = cerr_1/N_dist*100;
cerr_2 = cerr_2/N_dist*100;
cerr_3 = cerr_3/N_dist*100;

figure(1);
semilogx(t,err_3,t,cerr_3);
figure(2);
semilogx(t,err_2,t,cerr_2);
figure(3);
semilogx(t,err_1,t,cerr_1);
figure(4);
semilogx(t,err_0,t,cerr_0);
```

Appendix D

Simulation Stimulus Scripts and Waveforms

The scripts and waveforms in this appendix were used by, and generated with, respectively, the WaveGen tools to provide stimulus for the schematics in appendix B.

Listing D.1: Functional Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high          1.2
low      0.0


wave  sr1         low
wave  sr0         low
wave  pre         low
wave  din         low
wave  dout_en           low
vec   wl      4   0
vec   csr     4   0
vec   csw     4   0

// Begin here

// Write data to array
wl 4 at 5n

din low at 10n
csw 1 at 10n

csw 0 at 18n
din low at 19n
din high at 20n
csw 2 at 20n

csw 0 at 28n
din low at 29n
din high at 30n
csw 4 at 30n
csw 0 at 38n
din low at 39n
din low at 40n
csw 8 at 40n

csw 0 at 48n

wl 0 at 50n


// Read references
pre high at 55n
pre low at 70n

wl 4 at 75n
csr 1 at 80n
sr0 high at 80n

sr0 low at 88n
csr 0 at 89n
csr 2 at 90n
sr1 high at 90n

sr1 low at 98n
csr 0 at 99n


// Read data
csr 4 at 100n
dout_en high at 100n
csr 0 at 109n
csr 8 at 110n

csr 0 at 119n
wl 0 at 120n
dout_en low at 120n


end at 130n
```

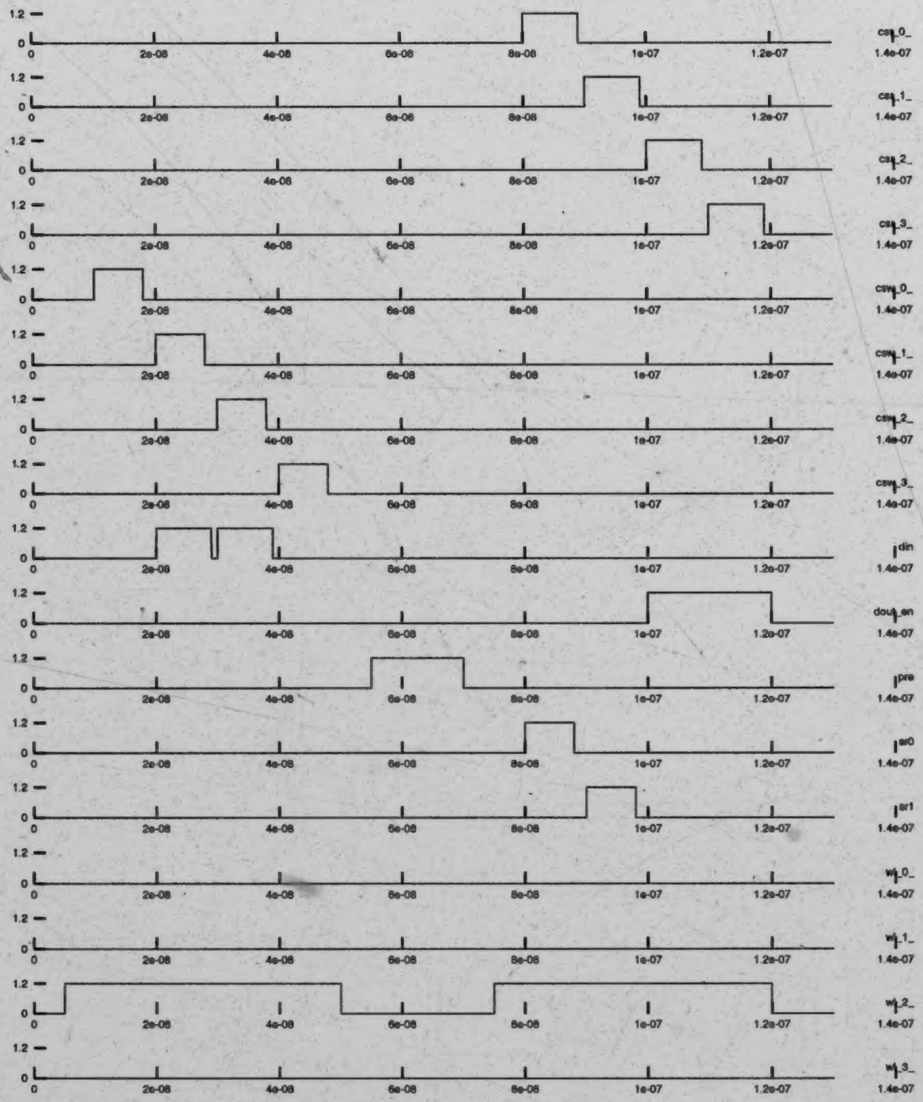


Figure D.1: Functional Simulation Stimulus Waveforms

Listing D.2: Coupling Measurement Stimulus WaveGen Script – No Coupling

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave WL4        low
wave WL3        low
wave WL2        low
wave WL1        low
wave WL0        low
wave PRE        low
wave Din        low
wave RCSEL4     low
wave RCSEL3     low
wave RCSEL2     low
wave RCSEL1     low
wave RCSEL0     low
wave WCSEL4     low
wave WCSEL3     low
wave WCSEL2     low
wave WCSEL1     low
wave WCSEL0     low


// No longer necessary to open row before precharge to reduce charge
// injection on writes, since 2 pass write scheme is used

// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din low at 35n
WCSEL0 1.8 at 40n
WCSEL0 low at 50n
Din low at 55n
WCSEL1 1.8 at 60n
WCSEL1 low at 70n
Din high at 75n
WCSEL2 1.8 at 80n
WCSEL2 low at 90n
Din low at 95n
WCSEL3 1.8 at 100n
WCSEL3 low at 110n
Din low at 115n
WCSEL4 1.8 at 120n
WCSEL4 low at 130n

// WRITE DATA PASS 2
Din low at 135n
WCSEL0 1.8 at 140n
WCSEL0 low at 150n
Din low at 155n
WCSEL1 1.8 at 160n
WCSEL1 low at 170n
Din high at 175n
WCSEL2 1.8 at 180n
WCSEL2 low at 190n
```

```
Din low at 195n
WCSEL3 1.8 at 200n
WCSEL3 low at 210n
Din low at 215n
WCSEL4 1.8 at 220n
WCSEL4 low at 230n

// CLOSE ROW
WL2 low at 240n

// PRECHARGE
PRE 1.8 at 250n
PRE low at 280n

// OPEN ROW FOR READ
WL2 1.8 at 290n

// READ DATA
RCSEL2 high at 300n
RCSEL2 low at 380n


// part 2 -- low measurement

// RE-PRECHARGE
PRE 1.8 at 390n
PRE low at 430n

// CLOSE ROW
WL2 low at 440n

// PRECHARGE
PRE 1.8 at 450n
PRE low at 480n

// OPEN ROW
WL2 1.8 at 485n

// WRITE DATA
Din high at 495n
WCSEL0 1.8 at 500n
WCSEL0 low at 510n
Din high at 515n
WCSEL1 1.8 at 520n
WCSEL1 low at 530n
Din low at 535n
WCSEL2 1.8 at 540n
WCSEL2 low at 550n
Din high at 555n
WCSEL3 1.8 at 560n
WCSEL3 low at 570n
Din high at 575n
WCSEL4 1.8 at 580n
WCSEL4 low at 590n

// WRITE DATA PASS 2
Din high at 595n
WCSEL0 1.8 at 600n
WCSEL0 low at 610n
Din high at 615n
WCSEL1 1.8 at 620n
WCSEL1 low at 630n
Din low at 635n
WCSEL2 1.8 at 640n
WCSEL2 low at 650n
Din high at 655n
```

```
WCSEL3 1.8 at 660n
WCSEL3 low at 670n
Din high at 675n
WCSEL4 1.8 at 680n
WCSEL4 low at 690n

// CLOSE ROW
WL2 low at 700n

// PRECHARGE
PRE 1.8 at 710n
PRE low at 740n

// OPEN ROW FOR READ
WL2 1.8 at 750n

// READ DATA
RCSEL2 high at 760n
RCSEL2 low at 840n

// CLOSE ROW
WL2 low at 850n


end at 860n
```

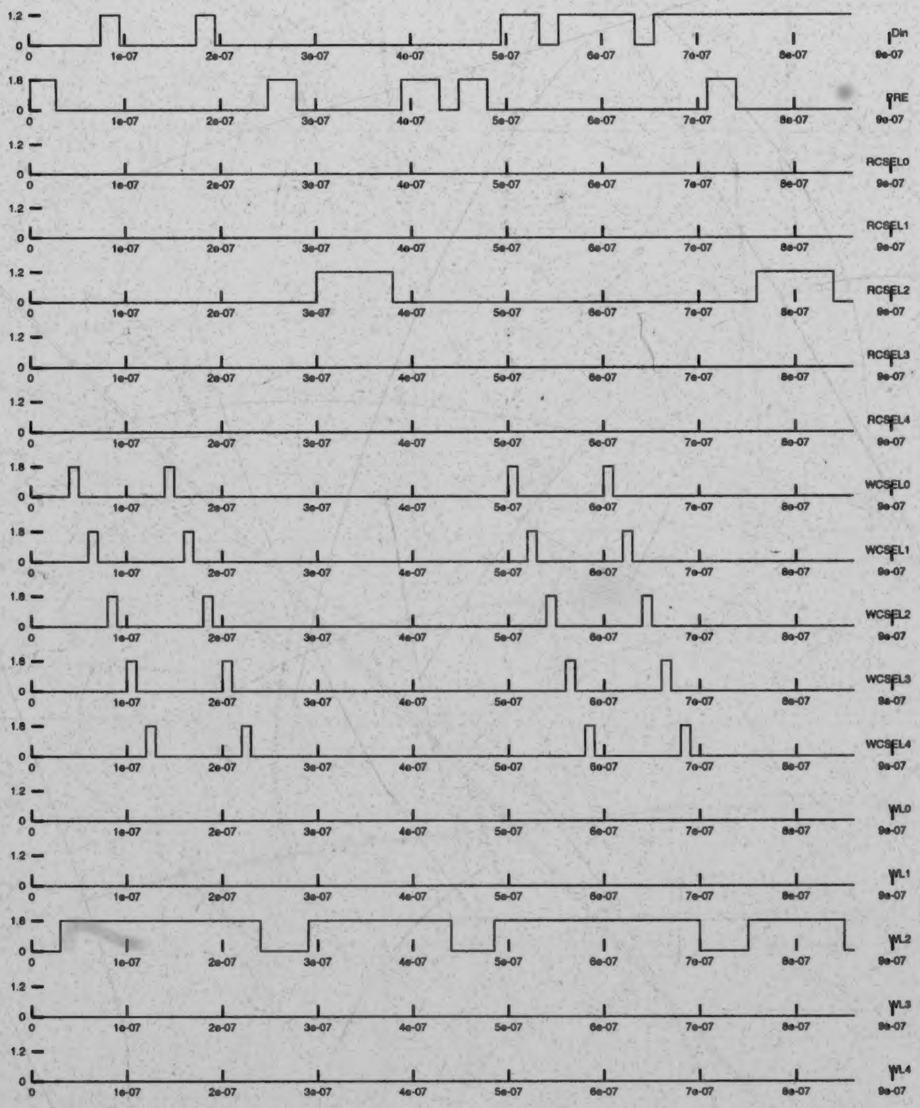


Figure D.2: Coupling Measurement Stimulus Waveforms – No Coupling

Listing D.3: Coupling Measurement Stimulus WaveGen Script – Worst Case '0'

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave WL4        low
wave WL3        low
wave WL2        low
wave WL1        low
wave WL0        low
wave PRE        low
wave Din        low
wave RCSEL4     low
wave RCSEL3     low
wave RCSEL2     low
wave RCSEL1     low
wave RCSEL0     low
wave WCSEL4     low
wave WCSEL3     low
wave WCSEL2     low
wave WCSEL1     low
wave WCSEL0     low


// No longer necessary to open row before precharge to reduce charge
// injection on writes, since 2 pass write scheme is used

// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din high at 35n
WCSEL0 1.8 at 40n
WCSEL0 low at 50n
Din high at 55n
WCSEL1 1.8 at 60n
WCSEL1 low at 70n
Din low at 75n
WCSEL2 1.8 at 80n
WCSEL2 low at 90n
Din high at 95n
WCSEL3 1.8 at 100n
WCSEL3 low at 110n
Din high at 115n
WCSEL4 1.8 at 120n
WCSEL4 low at 130n

// WRITE DATA PASS 2
Din high at 135n
WCSEL0 1.8 at 140n
WCSEL0 low at 150n
Din high at 155n
WCSEL1 1.8 at 160n
WCSEL1 low at 170n
Din low at 175n
WCSEL2 1.8 at 180n
WCSEL2 low at 190n
```

```
Din high at 195n
WCSEL3 1.8 at 200n
WCSEL3 low at 210n
Din high at 215n
WCSEL4 1.8 at 220n
WCSEL4 low at 230n

// CLOSE ROW
WL2 low at 240n

// PRECHARGE
PRE 1.8 at 250n
PRE low at 280n

// OPEN ROW FOR READ
WL2 1.8 at 290n

// READ DATA
RCSEL2 high at 300n
RCSEL2 low at 380n

// READ DATA ( all )
//RCSEL0 high at 300n
//RCSEL0 low at 390n
//RCSEL1 high at 400n
//RCSEL1 low at 490n
//RCSEL2 high at 500n
//RCSEL2 low at 590n
//RCSEL3 high at 600n
//RCSEL3 low at 690n
//RCSEL4 high at 700n
//RCSEL4 low at 790n

// CLOSE ROW
WL2 low at 390n


end at 400n
```

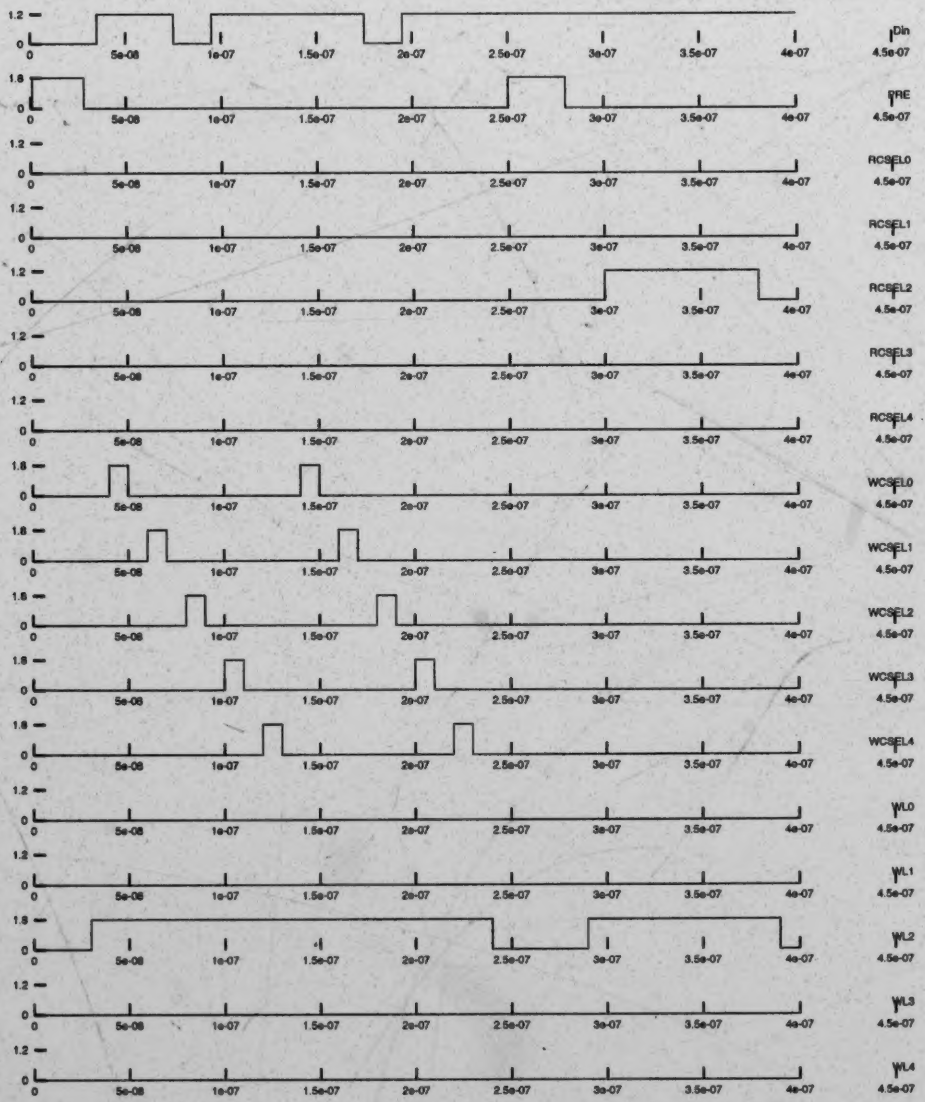


Figure D.3: Coupling Measurement Stimulus Waveforms – Worst Case '0'

Listing D.4: Coupling Measurement Stimulus WaveGen Script – Worst Case '1'

```
/* Wavegen – Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave WL4         low
wave WL3         low
wave WL2         low
wave WL1         low
wave WL0         low
wave PRE         low
wave Din         low
wave RCSEL4      low
wave RCSEL3      low
wave RCSEL2      low
wave RCSEL1      low
wave RCSEL0      low
wave WCSEL4      low
wave WCSEL3      low
wave WCSEL2      low
wave WCSEL1      low
wave WCSEL0      low


// No longer necessary to open row before precharge to reduce charge
// injection on writes, since 2 pass write scheme is used

// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din low at 35n
WCSEL0 1.8 at 40n
WCSEL0 low at 50n
Din low at 55n
WCSEL1 1.8 at 60n
WCSEL1 low at 70n
Din high at 75n
WCSEL2 1.8 at 80n
WCSEL2 low at 90n
Din low at 95n
WCSEL3 1.8 at 100n
WCSEL3 low at 110n
Din low at 115n
WCSEL4 1.8 at 120n
WCSEL4 low at 130n

// WRITE DATA PASS 2
Din low at 135n
WCSEL0 1.8 at 140n
WCSEL0 low at 150n
Din low at 155n
WCSEL1 1.8 at 160n
WCSEL1 low at 170n
Din high at 175n
WCSEL2 1.8 at 180n
WCSEL2 low at 190n
```

```
Din low at 195n
WCSEL3 1.8 at 200n
WCSEL3 low at 210n
Din low at 215n
WCSEL4 1.8 at 220n
WCSEL4 low at 230n

// CLOSE ROW
WL2 low at 240n

// PRECHARGE
PRE 1.8 at 250n
PRE low at 280n

// OPEN ROW FOR READ
WL2 1.8 at 290n

// READ DATA
RCSEL2 high at 300n
RCSEL2 low at 380n

// READ DATA ( all )
//RCSEL0 high at 300n
//RCSEL0 low at 390n
//RCSEL1 high at 400n
//RCSEL1 low at 490n
//RCSEL2 high at 500n
//RCSEL2 low at 590n
//RCSEL3 high at 600n
//RCSEL3 low at 690n
//RCSEL4 high at 700n
//RCSEL4 low at 790n

// CLOSE ROW
WL2 low at 390n


end at 400n
```

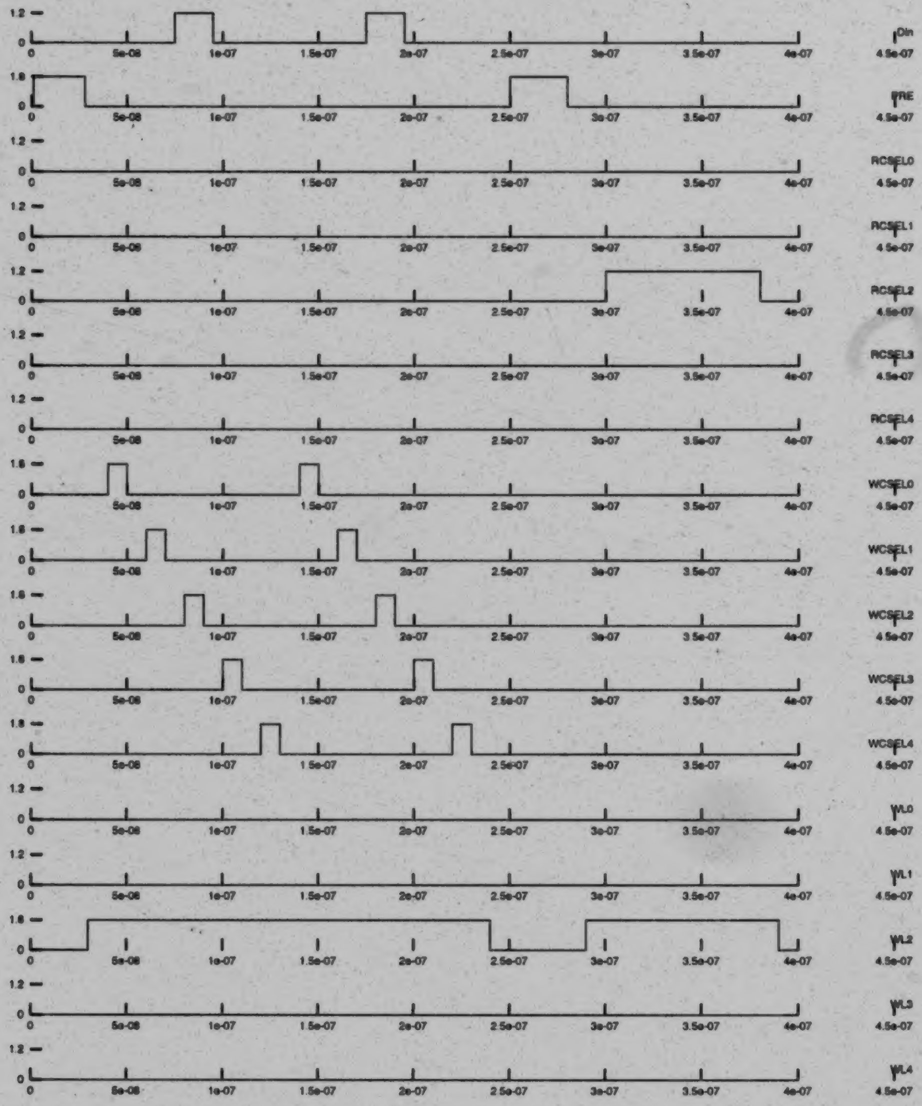


Figure D.4: Coupling Measurement Stimulus Waveforms – Worst Case '1'

Listing D.5: Performance Measurement Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low     0.0


wave WL4        low
wave WL3        low
wave WL2        low
wave WL1        low
wave WL0        low
wave PRE        low
wave Din        low
wave RCSEL4     low
wave RCSEL3     low
wave RCSEL2     low
wave RCSEL1     low
wave RCSEL0     low
wave WCSEL4     low
wave WCSEL3     low
wave WCSEL2     low
wave WCSEL1     low
wave WCSEL0     low


// First write: set up worst case write conditions
// Also, worst case precharge time measurement
// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din low at 35n
WCSEL1 1.8 at 40n
WCSEL1 low at 50n

Din high at 55n
WCSEL3 1.8 at 60n
WCSEL3 low at 70n

// CLOSE ROW
WL2 low at 80n

// Second write: write time measurement
// PRECHARGE
PRE 1.8 at 85n
PRE low at 110n

// OPEN ROW
WL2 1.8 at 120n

// WRITE DATA
Din high at 130n
WCSEL1 1.8 at 135n
WCSEL1 low at 145n

Din low at 150n
WCSEL3 1.8 at 155n
WCSEL3 low at 165n
```

```
// CLOSE ROW
WL2 low at 170n

// Read time measurement
// PRECHARGE
PRE 1.8 at 175n
PRE low at 200n

// OPEN ROW FOR READ
WL2 1.8 at 210n

// READ DATA
RCSEL1 high at 220n
RCSEL1 low at 310n
RCSEL3 high at 320n
RCSEL3 low at 400n

// CLOSE ROW
WL2 low at 410n


end at 420n
```

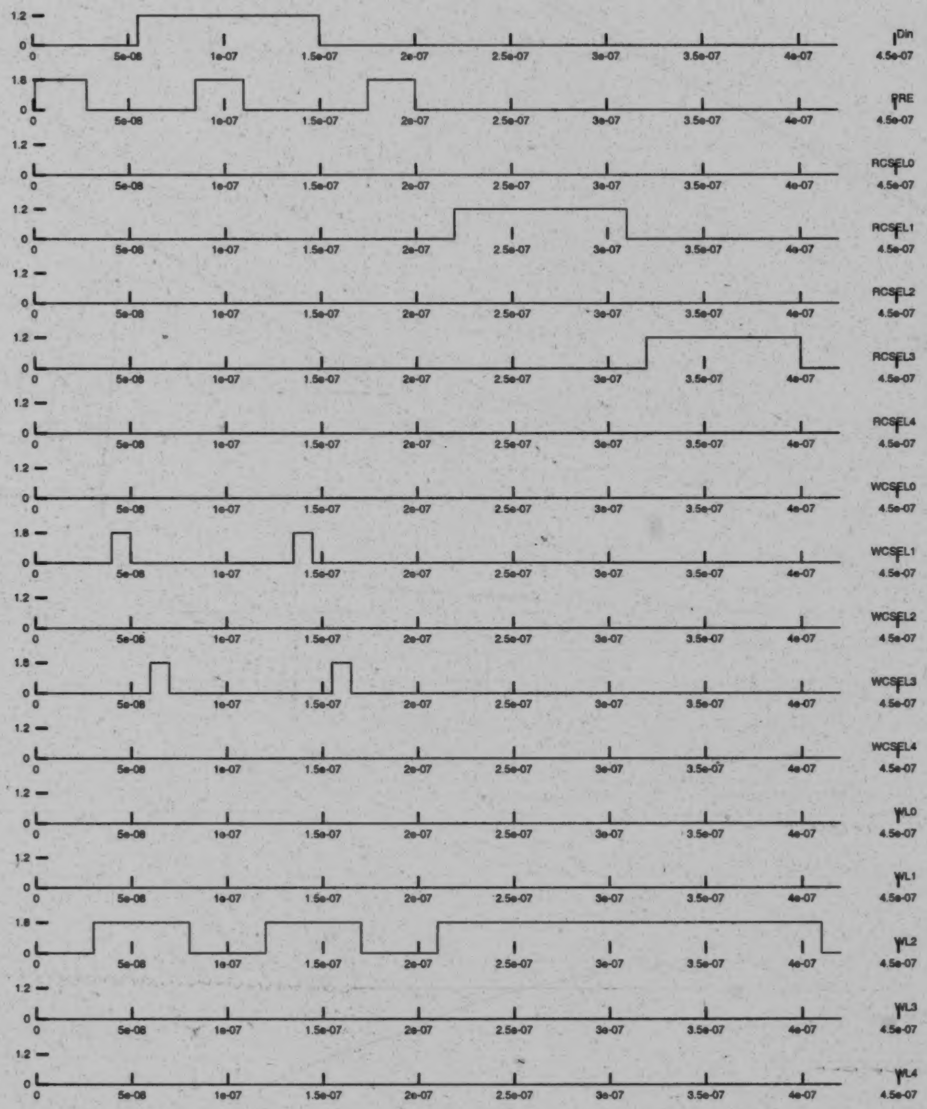


Figure D.5: Performance Measurement Stimulus Waveforms

Listing D.6: Bitline Power Measurement Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time   10p
fall_time   10p
high        1.2
low     0.0


wave WL4        low
wave WL3        low
wave WL2        low
wave WL1        low
wave WL0        low
wave PRE        low
wave Din        low
wave RCSEL4     low
wave RCSEL3     low
wave RCSEL2     low
wave RCSEL1     low
wave RCSEL0     low
wave WCSEL4     low
wave WCSEL3     low
wave WCSEL2     low
wave WCSEL1     low
wave WCSEL0     low


// First write -- prepare cells for read
// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din low at 35n
WCSEL1 1.8 at 40n
WCSEL1 low at 50n

Din high at 55n
WCSEL3 1.8 at 60n
WCSEL3 low at 70n

// CLOSE ROW
WL2 low at 80n

// Second write: write power measurement
// PRECHARGE
PRE 1.8 at 85n
PRE low at 110n

// OPEN ROW
WL2 1.8 at 120n

// WRITE DATA
Din high at 130n
WCSEL1 1.8 at 135n
WCSEL1 low at 145n

Din high at 150n
WCSEL3 1.8 at 155n
WCSEL3 low at 165n

// CLOSE ROW
```

```
WL2 low  at 170n

end  at 180n
```

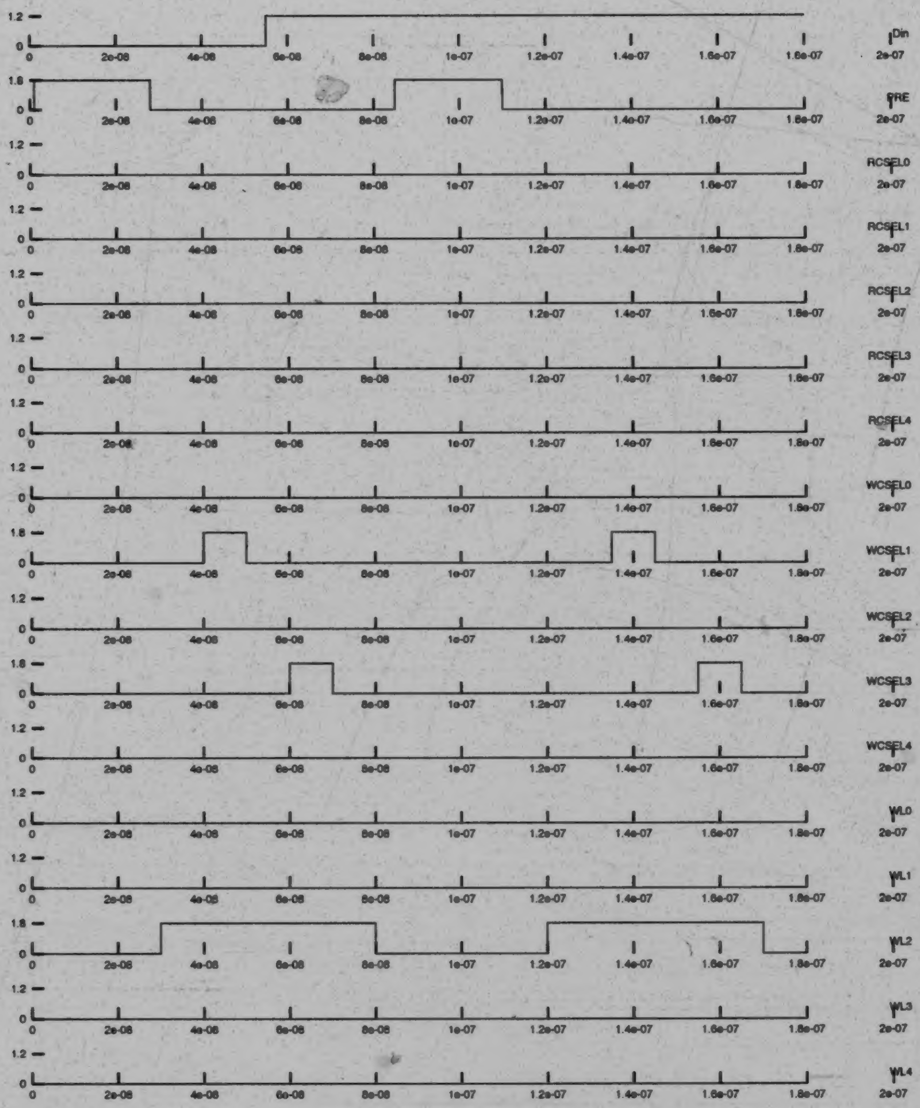


Figure D.6: Bitline Power Measurement Stimulus Waveforms

Listing D.7: Core Power Measurement Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave WL4       low
wave WL3       low
wave WL2       low
wave WL1       low
wave WL0       low
wave PRE       low
wave Din       low
wave RCSEL4    low
wave RCSEL3    low
wave RCSEL2    low
wave RCSEL1    low
wave RCSEL0    low
wave WCSEL4    low
wave WCSEL3    low
wave WCSEL2    low
wave WCSEL1    low
wave WCSEL0    low


/////
// Read bus power consumption
/////

// First write -- prepare cells for read
// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din low at 35n
WCSEL1 1.8 at 40n
WCSEL1 low at 50n

Din high at 55n
WCSEL3 1.8 at 60n
WCSEL3 low at 70n

// CLOSE ROW
WL2 low at 80n

// Read bus power measurement
// PRECHARGE
PRE 1.8 at 85n
PRE low at 110n

// OPEN ROW
WL2 1.8 at 120n

// READ DATA
RCSEL1 1.2 at 130n
RCSEL1 low at 220n

RCSEL3 1.2 at 230n
RCSEL3 low at 320n
```

```
// CLOSE ROW
WL2 low  at 330n

end  at 340n
```

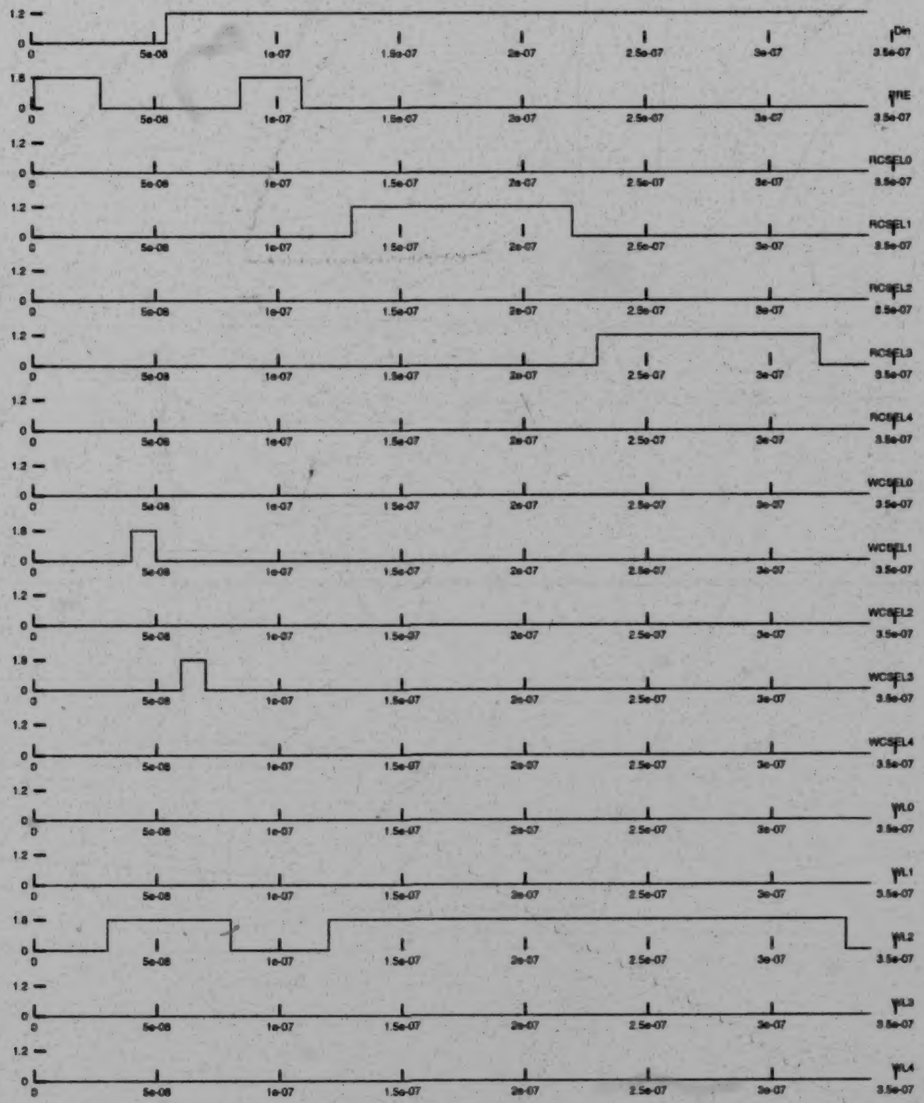


Figure D.7: Core Power Measurement Stimulus Waveforms

Listing D.8: Wine Cellar Technique Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave WL4        -0.5
wave WL3        -0.5
wave WL2        -0.5
wave WL1        -0.5
wave WL0        -0.5
wave PRE        low
wave Din        low
wave RCSEL4     low
wave RCSEL3     low
wave RCSEL2     low
wave RCSEL1     low
wave RCSEL0     low
wave WCSEL4     low
wave WCSEL3     low
wave WCSEL2     low
wave WCSEL1     low
wave WCSEL0     low



// PRECHARGE
PRE 1.8 at 1n
PRE low at 28n

// OPEN ROW
WL2 1.8 at 30n

// WRITE DATA
Din high at 35n
WCSEL0 1.8 at 40n
WCSEL0 low at 50n
Din 0.4 at 55n
WCSEL1 1.8 at 60n
WCSEL1 low at 70n
Din high at 75n
WCSEL2 1.8 at 80n
WCSEL2 low at 90n
Din 0.8 at 95n
WCSEL3 1.8 at 100n
WCSEL3 low at 110n
Din high at 115n
WCSEL4 1.8 at 120n
WCSEL4 low at 130n

// WRITE DATA PASS 2
Din high at 135n
WCSEL0 1.8 at 140n
WCSEL0 low at 150n
Din 0.4 at 155n
WCSEL1 1.8 at 160n
WCSEL1 low at 170n
Din high at 175n
WCSEL2 1.8 at 180n
WCSEL2 low at 190n
Din 0.8 at 195n
WCSEL3 1.8 at 200n
WCSEL3 low at 210n
```

```
Din high at 215n
WCSEL4 1.8 at 220n
WCSEL4 low at 230n

// CLOSE ROW
WL2 -0.5 at 240n

// PRECHARGE
PRE 1.8 at 250n
PRE low at 280n


end at 1
```

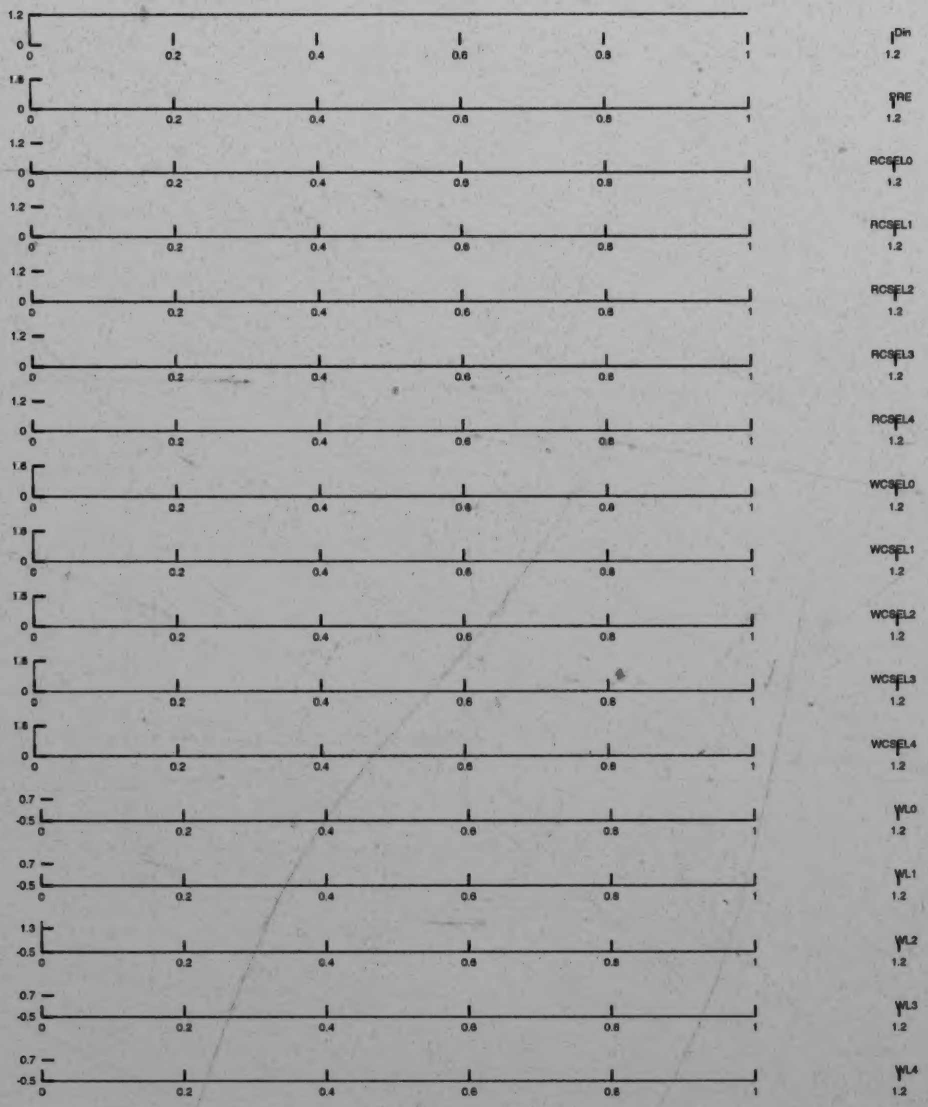


Figure D.8: Wine Cellar Technique Simulation Stimulus Waveforms

Listing D.9: Current Mirror Amplifier Transient Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time   10p
fall_time   10p
high        1.2
low     0.0


wave BL      high
wave RCSEL       high
wave RCcharge        high

// Begin here
RCSEL low at 50n
RCcharge low at 50n
RCcharge high at 55n

RCSEL high at 150n

end at 250n
```

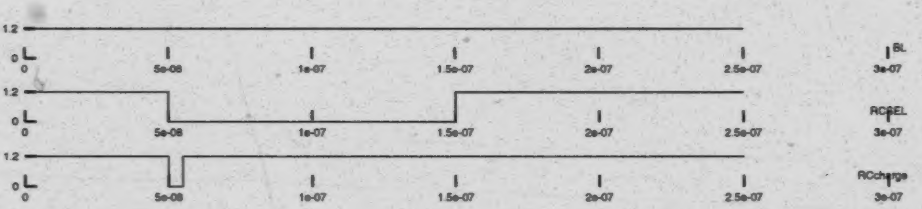


Figure D.9: Current Mirror Amplifier Transient Simulation Stimulus Waveforms

Listing D.10: Bus Precharge Amplifier I/O Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high          1.2
low      0.0


wave BL           high
wave RCSEL        high
wave PRE_BUS           high

// Begin here
RCSEL low at 10n

PRE_BUS low at 35n
PRE_BUS high at 45n

RCSEL high at 50n

end at 150n
```

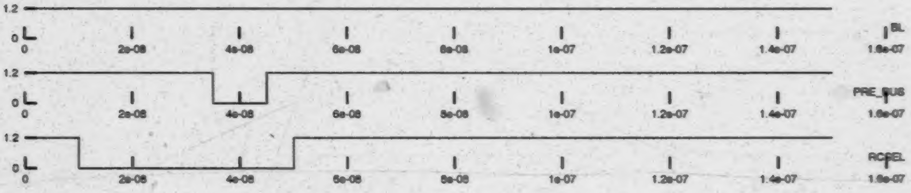


Figure D.10: Bus Precharge Amplifier I/O Simulation Stimulus Waveforms

Listing D.11: Bus Precharge Amplifier Transient Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time    10p
fall_time    10p
high         1.2
low      0.0


wave BL          high
wave RCSEL       high
wave PRE_BUS          high

// Begin here
RCSEL low at 10n

PRE_BUS low at 35n
PRE_BUS high at 45n

RCSEL high at 50n


RCSEL low at 145n

BL low at 150n

PRE_BUS low at 150n
PRE_BUS high at 160n

RCSEL high at 165n

end at 260n
```

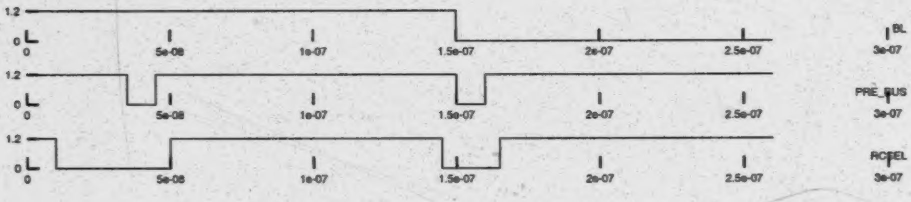


Figure D.11: Bus Precharge Amplifier Transient Simulation Stimulus Waveforms

Listing D.12: Multilevel Functional Simulation Stimulus WaveGen Script

```
/* Wavegen - Waveform Generation Scripting Language */

rise_time     10p
fall_time     10p
high          1.2
low      0.0

wave  sr3          low
wave  sr2          low
wave  sr1          low
wave  sr0          low
wave  pre          low
wave  din          low
wave  dout_en             low
vec   wl      8    0
vec   csr     8    0
vec   csw     8    0

// Begin here

// Write data to array
wl 4 at 5n

din 0.0 at 10n
csw 1 at 10n

csw 0 at 18n
din 0.0 at 19n
din 0.4 at 20n
csw 2 at 20n

csw 0 at 28n
din 0.0 at 29n
din 0.8 at 30n
csw 4 at 30n

csw 0 at 38n
din 0.0 at 39n
din 1.2 at 40n
csw 8 at 40n

csw 0 at 48n
din 0.0 at 49n
din 0.4 at 50n
csw 16 at 50n

csw 0 at 58n
din 0.0 at 59n
din 1.2 at 60n
csw 32 at 60n

csw 0 at 68n
din 0.0 at 69n
din 0.0 at 70n
csw 64 at 70n

csw 0 at 78n
din 0.0 at 79n
din 0.8 at 80n
csw 128 at 80n

csw 0 at 88n
din 0.0 at 89n

wl 0 at 90n
```

```
// Read references
pre high at 95n
pre low at 110n

wl 4 at 115n
csr 1 at 120n
sr0 high at 120n

sr0 low at 128n
csr 0 at 129n
csr 2 at 130n
sr1 high at 130n

sr1 low at 138n
csr 0 at 139n
csr 4 at 140n
sr2 high at 140n

sr2 low at 148n
csr 0 at 149n
csr 8 at 150n
sr3 high at 150n

sr3 low at 158n
csr 0 at 159n


// Read data
dout_en high at 160n
csr 16 at 160n

csr 0 at 169n
csr 32 at 170n

csr 0 at 179n
csr 64 at 180n

csr 0 at 189n
csr 128 at 190n

csr 0 at 199n
wl 0 at 200n
dout_en low at 200n


end at 210n
```

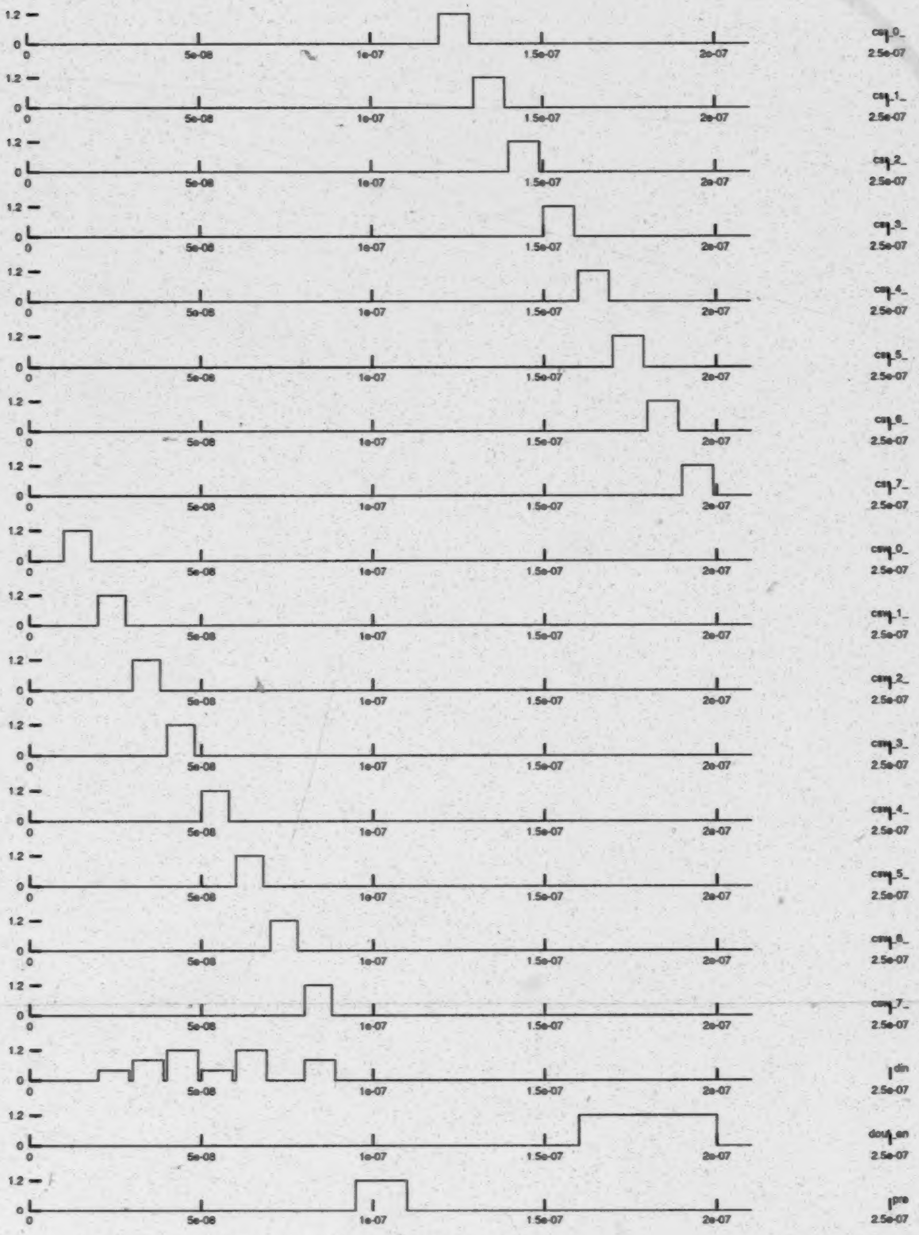


Figure D.12: Multilevel Functional Simulation Stimulus Waveforms - Page 1

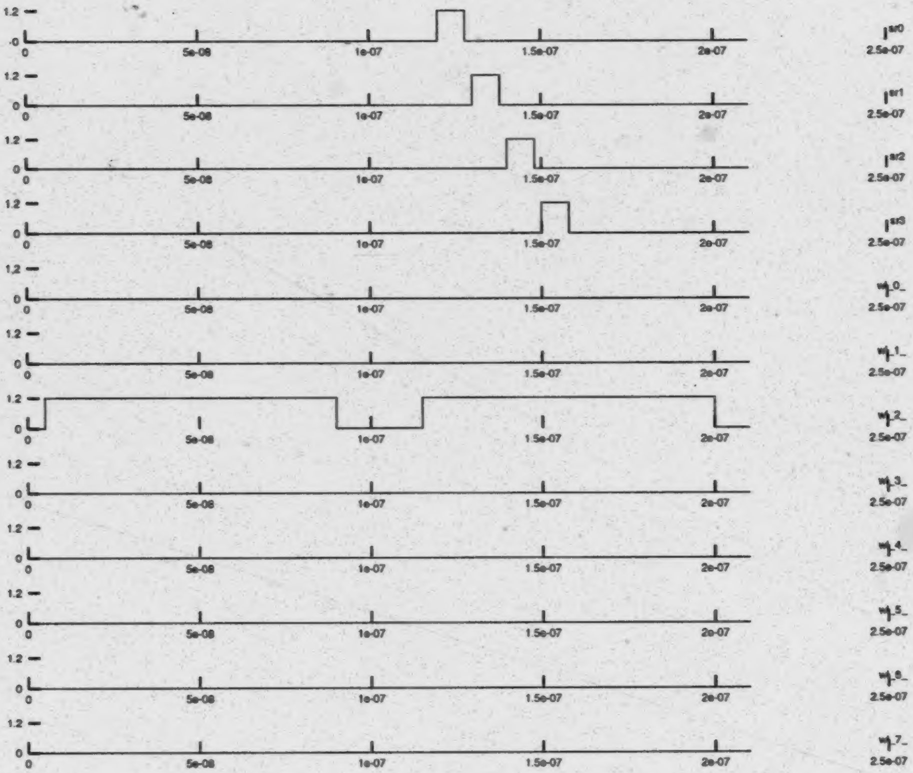


Figure D.13: Multilevel Functional Simulation Stimulus Waveforms - Page 2

Appendix E

Verilog-A HDL Code

The following HDL code was written to verify the functionality of the core of the proposed architecture. Also included in this appendix is a schematic of the interconnection of the HDL modules for simulation.

Listing E.1: Cell Array Code (4x4)

```
// VerilogA for func_sim, func_array, veriloga
// This module implements a 4x4 functional DRAM array

`include "constants.h"
`include "discipline.h"

module func_bl(wl0, wl1, wl2, wl3, bl0, bl1, bl2, bl3);
input wl0, wl1, wl2, wl3;  // Wordlines
inout bl0, bl1, bl2, bl3;  // Bitline input/output terminals
electrical wl0, wl1, wl2, wl3;
electrical bl0, bl1, bl2, bl3;

electrical cell00, cell01, cell02, cell03;
electrical cell10, cell11, cell12, cell13;
electrical cell20, cell21, cell22, cell23;
electrical cell30, cell31, cell32, cell33;

parameter real Cs = 25f;
parameter real Cb = 100f;
parameter real V_on = 0.6;
parameter real R_trans = 4k;
parameter real G_bl = 10f;


analog begin
  @ ( initial_step ) begin

    // Initialize cells
    V(cell00) <+ 0.0;
    V(cell01) <+ 0.0;
    V(cell02) <+ 0.0;
    V(cell03) <+ 0.0;

    V(cell10) <+ 0.0;
    V(cell11) <+ 0.0;
    V(cell12) <+ 0.0;
    V(cell13) <+ 0.0;
  V(cell20) <+ 0.0;
  V(cell21) <+ 0.0;
  V(cell22) <+ 0.0;
  V(cell23) <+ 0.0;

  V(cell30) <+ 0.0;
  V(cell31) <+ 0.0;
  V(cell32) <+ 0.0;
  V(cell33) <+ 0.0;
end

// Define cell capacitances
I(cell00) <+ Cs * ddt(V(cell00));
I(cell01) <+ Cs * ddt(V(cell01));
I(cell02) <+ Cs * ddt(V(cell02));
I(cell03) <+ Cs * ddt(V(cell03));

I(cell10) <+ Cs * ddt(V(cell10));
I(cell11) <+ Cs * ddt(V(cell11));
I(cell12) <+ Cs * ddt(V(cell12));
I(cell13) <+ Cs * ddt(V(cell13));

I(cell20) <+ Cs * ddt(V(cell20));
I(cell21) <+ Cs * ddt(V(cell21));
I(cell22) <+ Cs * ddt(V(cell22));
I(cell23) <+ Cs * ddt(V(cell23));

I(cell30) <+ Cs * ddt(V(cell30));
I(cell31) <+ Cs * ddt(V(cell31));
I(cell32) <+ Cs * ddt(V(cell32));
I(cell33) <+ Cs * ddt(V(cell33));

// Define bitline capacitances
I(bl0) <+ Cb * ddt(V(bl0));
I(bl1) <+ Cb * ddt(V(bl1));
I(bl2) <+ Cb * ddt(V(bl2));
I(bl3) <+ Cb * ddt(V(bl3));

// Define cell access
if (V(wl0) > V_on) begin
  I(bl0, cell00) <+ V(bl0, cell00) / R_trans;
  I(bl1, cell10) <+ V(bl1, cell10) / R_trans;
  I(bl2, cell20) <+ V(bl2, cell20) / R_trans;
  I(bl3, cell30) <+ V(bl3, cell30) / R_trans;
end
else begin
  I(bl0, cell00) <+ 0.0;
  I(bl1, cell10) <+ 0.0;
  I(bl2, cell20) <+ 0.0;
  I(bl3, cell30) <+ 0.0;
end

if (V(wl1) > V_on) begin
  I(bl0, cell01) <+ V(bl0, cell01) / R_trans;
  I(bl1, cell11) <+ V(bl1, cell11) / R_trans;
  I(bl2, cell21) <+ V(bl2, cell21) / R_trans;
  I(bl3, cell31) <+ V(bl3, cell31) / R_trans;
end
else begin
  I(bl0, cell01) <+ 0.0;
  I(bl1, cell11) <+ 0.0;
  I(bl2, cell21) <+ 0.0;
  I(bl3, cell31) <+ 0.0;
end

if (V(wl2) > V_on) begin
```

```
    I(bl0 , cell02 ) <+ V(bl0 , cell02 ) / R_trans ;
    I(bl1 , cell12 ) <+ V(bl1 , cell12 ) / R_trans ;
    I(bl2 , cell22 ) <+ V(bl2 , cell22 ) / R_trans ;
    I(bl3 , cell32 ) <+ V(bl3 , cell32 ) / R_trans ;
  end
  else begin
    I(bl0 , cell02 ) <+ 0.0;
    I(bl1 , cell12 ) <+ 0.0;
    I(bl2 , cell22 ) <+ 0.0;
    I(bl3 , cell32 ) <+ 0.0;
  end

  if (V(wl3) > V_on) begin
    I(bl0 , cell03 ) <+ V(bl0 , cell03 ) / R_trans ;
    I(bl1 , cell13 ) <+ V(bl1 , cell13 ) / R_trans ;
    I(bl2 , cell23 ) <+ V(bl2 , cell23 ) / R_trans ;
    I(bl3 , cell33 ) <+ V(bl3 , cell33 ) / R_trans ;
  end
  else begin
    I(bl0 , cell03 ) <+ 0.0;
    I(bl1 , cell13 ) <+ 0.0;
    I(bl2 , cell23 ) <+ 0.0;
    I(bl3 , cell33 ) <+ 0.0;
  end

  // Install bl conductance to aid Spectre
  I(bl0) <+ V(bl0) * G_bl;
  I(bl1) <+ V(bl1) * G_bl;
  I(bl2) <+ V(bl2) * G_bl;
  I(bl3) <+ V(bl3) * G_bl;

end


endmodule
```

Listing E.2: Sense Amplifier Code

```
// VerilogA for memory, func_sense, veriloga
// This module implements a functional analog sense amplifier

‘include ”constants.h”
‘include ”discipline.h”

module func_sense(bl, vpre, din, pre, csw, csr, iout);
input bl, vpre, din, pre, csw, csr;
inout iout;
electrical bl, vpre; din, pre, csw, csr, iout;

parameter real V_th = 0.45;
parameter real Gain = 80u;
parameter real R_trans = 4k;

analog begin

  if ((V(bl) < V_th) || (V(csr) < V_th)) begin
    I(iout) <+ 0.0;
  end
  else begin
    I(iout) <+ Gain * (V(bl)-V_th);
  end

  if (V(pre) < V_th) begin
    I(bl, vpre) <+ 0.0;
  end
  else begin
```

```
    I(bl, vpre) <+ V(bl, vpre) / R_trans;
  end

  if (V(csw) < V_th) begin
    I(bl, din) <+ 0.0;
  end
  else begin
    I(bl, din) <+ V(bl, din) / R_trans;
  end

end


endmodule
```

Listing E.3: Bus Amplifier Code

```
// VerilogA for memory, func_pa, veriloga
// This module implements a functional primary amplifier

‘include "constants.h"
‘include "discipline.h"

module func_pa(ibus, vout);
input ibus;
output vout;
electrical ibus, vout;

parameter real pa_Gain = 20k;

analog begin

    V(vout) <+ -(pa_Gain * I(ibus));

end


endmodule
```

Listing E.4: Data Converter Code

```
// VerilogA for func_adc

‘include "discipline.h"
‘include "constants.h"

module func_adc(sr0, sr1, en, vin, dout);
input en, vin;
output dout;
electrical sr0, sr1, en, vin, dout;

parameter real V_th = 0.45;
parameter real Vdd = 1.2;

real ref0, ref1;
real dout_val;

  analog begin
    @ ( initial_step ) begin
      V(dout) <+ 0.0;
      I(dout) <+ 0.0;
      ref0 = 0.0;
      ref1 = 0.0;
      dout_val = 0.0;
    end
    if ((V(sr0) > V_th) && (V(sr1) < V_th)) begin
      ref0 = V(vin);
    end
    else if ((V(sr1) > V_th) && (V(sr0) < V_th)) begin
      ref1 = V(vin);
    end
    else if ((V(sr0) < V_th) && (V(sr1) < V_th)) begin
      if ((V(vin) - ref0) < (ref1 - V(vin))) begin
        dout_val = 0.0;
      end
      else begin
        dout_val = Vdd;
      end
    end

    if (V(en) > V_th) begin
      V(dout) <+ dout_val;
    end
    else begin
      V(dout) <+ 0.0;
    end

  end

endmodule
```

Listing E.5: Cell Array Code (8x8)

```
// VerilogA for func_sim, func_array, veriloga
// This module implements a 8x8 functional DRAM array

`include "constants.h"
`include "discipline.h"

module func_bl(wl0, wl1, wl2, wl3, wl4, wl5, wl6, wl7,
               bl0, bl1, bl2, bl3, bl4, bl5, bl6, bl7);

input wl0, wl1, wl2, wl3, wl4, wl5, wl6, wl7;  // Wordlines
inout bl0, bl1, bl2, bl3, bl4, bl5, bl6, bl7;  // Bitline input/output terminals
electrical wl0, wl1, wl2, wl3, wl4, wl5, wl6, wl7;
electrical bl0, bl1, bl2, bl3, bl4, bl5, bl6, bl7;

electrical cell00, cell01, cell02, cell03, cell04, cell05, cell06, cell07;
electrical cell10, cell11, cell12, cell13, cell14, cell15, cell16, cell17;
electrical cell20, cell21, cell22, cell23, cell24, cell25, cell26, cell27;
electrical cell30, cell31, cell32, cell33, cell34, cell35, cell36, cell37;
electrical cell40, cell41, cell42, cell43, cell44, cell45, cell46, cell47;
electrical cell50, cell51, cell52, cell53, cell54, cell55, cell56, cell57;
electrical cell60, cell61, cell62, cell63, cell64, cell65, cell66, cell67;
electrical cell70, cell71, cell72, cell73, cell74, cell75, cell76, cell77;

parameter real Cs = 25f;
parameter real Cb = 100f;
parameter real V_on = 0.6;
parameter real R_trans = 4k;
parameter real G_bl = 10f;


analog begin
  @ ( initial_step ) begin

    // Initialize cells
    V(cell00) <+ 0.0; V(cell01) <+ 0.0; V(cell02) <+ 0.0; V(cell03) <+ 0.0;
```

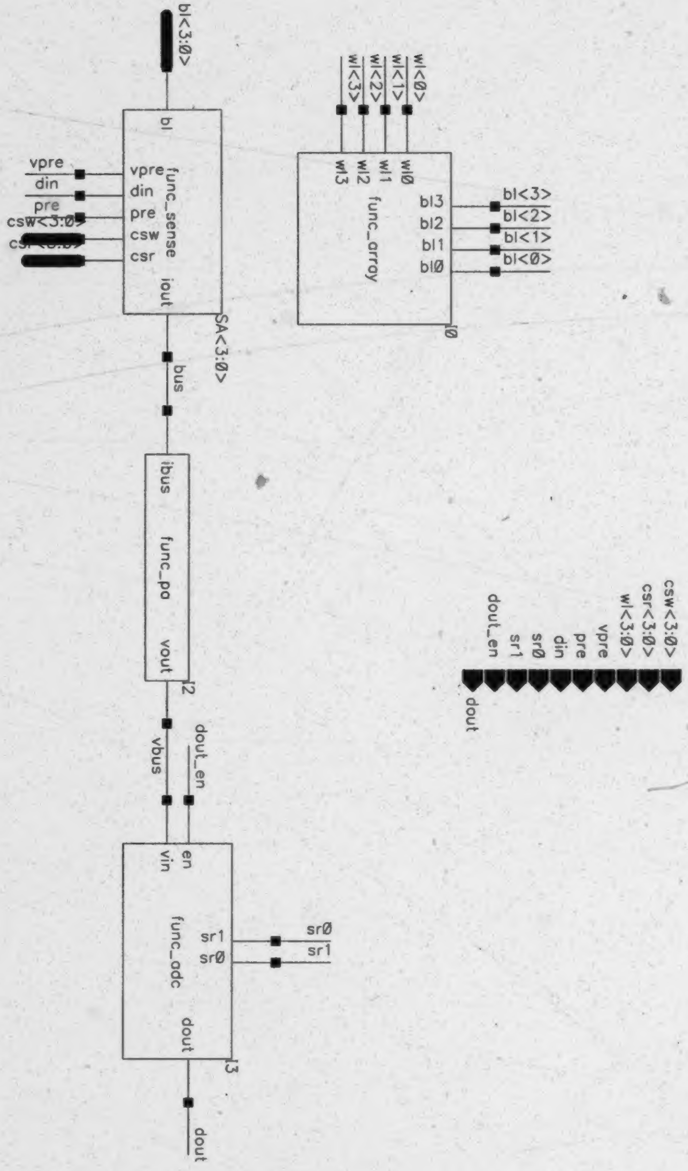

bl<3:0>
bl
func_sense
vpre
din
pre
csw
csr
iout
SA<3:0>
vpre
din
pre
csw<3:0>
csr<3:0>
bus
ibus
func_pa
vout
I2
vbus
dout_en
en
vin
func_adc
sr1
sr0
dout
I3
sr0
sr1
dout
wl<0>
wl<1>
wl<2>
wl<3>
wl0
wl1
wl2
wl3
func_array
bl3
bl2
bl1
bl0
bl<3>
bl<2>
bl<1>
bl<0>
I0
csw<3:0>
csr<3:0>
wl<3:0>
vpre
pre
din
sr0
sr1
dout_en
dout

Figure E.1: Top Level Interconnection Schematic for Functional Simulation

```
  V(cell04) <+ 0.0; V(cell05) <+ 0.0; V(cell06) <+ 0.0; V(cell07) <+ 0.0;

  V(cell10) <+ 0.0; V(cell11) <+ 0.0; V(cell12) <+ 0.0; V(cell13) <+ 0.0;
  V(cell14) <+ 0.0; V(cell15) <+ 0.0; V(cell16) <+ 0.0; V(cell17) <+ 0.0;

  V(cell20) <+ 0.0; V(cell21) <+ 0.0; V(cell22) <+ 0.0; V(cell23) <+ 0.0;
  V(cell24) <+ 0.0; V(cell25) <+ 0.0; V(cell26) <+ 0.0; V(cell27) <+ 0.0;

  V(cell30) <+ 0.0; V(cell31) <+ 0.0; V(cell32) <+ 0.0; V(cell33) <+ 0.0;
  V(cell34) <+ 0.0; V(cell35) <+ 0.0; V(cell36) <+ 0.0; V(cell37) <+ 0.0;

  V(cell40) <+ 0.0; V(cell41) <+ 0.0; V(cell42) <+ 0.0; V(cell43) <+ 0.0;
  V(cell44) <+ 0.0; V(cell45) <+ 0.0; V(cell46) <+ 0.0; V(cell47) <+ 0.0;

  V(cell50) <+ 0.0; V(cell51) <+ 0.0; V(cell52) <+ 0.0; V(cell53) <+ 0.0;
  V(cell54) <+ 0.0; V(cell55) <+ 0.0; V(cell56) <+ 0.0; V(cell57) <+ 0.0;

  V(cell60) <+ 0.0; V(cell61) <+ 0.0; V(cell62) <+ 0.0; V(cell63) <+ 0.0;
  V(cell64) <+ 0.0; V(cell65) <+ 0.0; V(cell66) <+ 0.0; V(cell67) <+ 0.0;

  V(cell70) <+ 0.0; V(cell71) <+ 0.0; V(cell72) <+ 0.0; V(cell73) <+ 0.0;
  V(cell74) <+ 0.0; V(cell75) <+ 0.0; V(cell76) <+ 0.0; V(cell77) <+ 0.0;
end


// Define cell capacitances
I(cell00) <+ Cs * ddt(V(cell00)); I(cell01) <+ Cs * ddt(V(cell01)); I(cell02)
    <+ Cs * ddt(V(cell02));
I(cell03) <+ Cs * ddt(V(cell03)); I(cell04) <+ Cs * ddt(V(cell04)); I(cell05)
    <+ Cs * ddt(V(cell05));
I(cell06) <+ Cs * ddt(V(cell06)); I(cell07) <+ Cs * ddt(V(cell07));

I(cell10) <+ Cs * ddt(V(cell10)); I(cell11) <+ Cs * ddt(V(cell11)); I(cell12)
    <+ Cs * ddt(V(cell12));
I(cell13) <+ Cs * ddt(V(cell13)); I(cell14) <+ Cs * ddt(V(cell14)); I(cell15)
    <+ Cs * ddt(V(cell15));
I(cell16) <+ Cs * ddt(V(cell16)); I(cell17) <+ Cs * ddt(V(cell17));

I(cell20) <+ Cs * ddt(V(cell20)); I(cell21) <+ Cs * ddt(V(cell21)); I(cell22)
    <+ Cs * ddt(V(cell22));
I(cell23) <+ Cs * ddt(V(cell23)); I(cell24) <+ Cs * ddt(V(cell24)); I(cell25)
    <+ Cs * ddt(V(cell25));
I(cell26) <+ Cs * ddt(V(cell26)); I(cell27) <+ Cs * ddt(V(cell27));

I(cell30) <+ Cs * ddt(V(cell30)); I(cell31) <+ Cs * ddt(V(cell31)); I(cell32)
    <+ Cs * ddt(V(cell32));
I(cell33) <+ Cs * ddt(V(cell33)); I(cell34) <+ Cs * ddt(V(cell34)); I(cell35)
    <+ Cs * ddt(V(cell35));
I(cell36) <+ Cs * ddt(V(cell36)); I(cell37) <+ Cs * ddt(V(cell37));

I(cell40) <+ Cs * ddt(V(cell40)); I(cell41) <+ Cs * ddt(V(cell41)); I(cell42)
    <+ Cs * ddt(V(cell42));
I(cell43) <+ Cs * ddt(V(cell43)); I(cell44) <+ Cs * ddt(V(cell44)); I(cell45)
    <+ Cs * ddt(V(cell45));
I(cell46) <+ Cs * ddt(V(cell46)); I(cell47) <+ Cs * ddt(V(cell47));

I(cell50) <+ Cs * ddt(V(cell50)); I(cell51) <+ Cs * ddt(V(cell51)); I(cell52)
    <+ Cs * ddt(V(cell52));
I(cell53) <+ Cs * ddt(V(cell53)); I(cell54) <+ Cs * ddt(V(cell54)); I(cell55)
    <+ Cs * ddt(V(cell55));
I(cell56) <+ Cs * ddt(V(cell56)); I(cell57) <+ Cs * ddt(V(cell57));

I(cell60) <+ Cs * ddt(V(cell60)); I(cell61) <+ Cs * ddt(V(cell61)); I(cell62)
    <+ Cs * ddt(V(cell62));
I(cell63) <+ Cs * ddt(V(cell63)); I(cell64) <+ Cs * ddt(V(cell64)); I(cell65)
    <+ Cs * ddt(V(cell65));
```

```
I(cell66) <+ Cs * ddt(V(cell66)); I(cell67) <+ Cs * ddt(V(cell67));

I(cell70) <+ Cs * ddt(V(cell70)); I(cell71) <+ Cs * ddt(V(cell71)); I(cell72)
     <+ Cs * ddt(V(cell72));
I(cell73) <+ Cs * ddt(V(cell73)); I(cell74) <+ Cs * ddt(V(cell74)); I(cell75)
     <+ Cs * ddt(V(cell75));
I(cell76) <+ Cs * ddt(V(cell76)); I(cell77) <+ Cs * ddt(V(cell77));


// Define bitline capacitances
I(bl0) <+ Cb * ddt(V(bl0));
I(bl1) <+ Cb * ddt(V(bl1));
I(bl2) <+ Cb * ddt(V(bl2));
I(bl3) <+ Cb * ddt(V(bl3));
I(bl4) <+ Cb * ddt(V(bl4));
I(bl5) <+ Cb * ddt(V(bl5));
I(bl6) <+ Cb * ddt(V(bl6));
I(bl7) <+ Cb * ddt(V(bl7));


// Define cell access
if (V(wl0) > V_on) begin
  I(bl0,cell00) <+ V(bl0,cell00) / R_trans; I(bl1,cell10) <+ V(bl1,cell10) /
      R_trans;
  I(bl2,cell20) <+ V(bl2,cell20) / R_trans; I(bl3,cell30) <+ V(bl3,cell30) /
      R_trans;
  I(bl4,cell40) <+ V(bl4,cell40) / R_trans; I(bl5,cell50) <+ V(bl5,cell50) /
      R_trans;
  I(bl6,cell60) <+ V(bl6,cell60) / R_trans; I(bl7,cell70) <+ V(bl7,cell70) /
      R_trans;
end
else begin
  I(bl0,cell00) <+ 0.0; I(bl1,cell10) <+ 0.0; I(bl2,cell20) <+ 0.0; I(bl3,cell30
      ) <+ 0.0;
  I(bl4,cell40) <+ 0.0; I(bl5,cell50) <+ 0.0; I(bl6,cell60) <+ 0.0; I(bl7,cell70
      ) <+ 0.0;
end

if (V(wl1) > V_on) begin
  I(bl0,cell01) <+ V(bl0,cell01) / R_trans; I(bl1,cell11) <+ V(bl1,cell11) /
      R_trans;
  I(bl2,cell21) <+ V(bl2,cell21) / R_trans; I(bl3,cell31) <+ V(bl3,cell31) /
      R_trans;
  I(bl4,cell41) <+ V(bl4,cell41) / R_trans; I(bl5,cell51) <+ V(bl5,cell51) /
      R_trans;
  I(bl6,cell61) <+ V(bl6,cell61) / R_trans; I(bl7,cell71) <+ V(bl7,cell71) /
      R_trans;
end
else begin
  I(bl0,cell01) <+ 0.0; I(bl1,cell11) <+ 0.0; I(bl2,cell21) <+ 0.0; I(bl3,cell31
      ) <+ 0.0;
  I(bl4,cell41) <+ 0.0; I(bl5,cell51) <+ 0.0; I(bl6,cell61) <+ 0.0; I(bl7,cell71
      ) <+ 0.0;
end

if (V(wl2) > V_on) begin
  I(bl0,cell02) <+ V(bl0,cell02) / R_trans; I(bl1,cell12) <+ V(bl1,cell12) /
      R_trans;
  I(bl2,cell22) <+ V(bl2,cell22) / R_trans; I(bl3,cell32) <+ V(bl3,cell32) /
      R_trans;
  I(bl4,cell42) <+ V(bl4,cell42) / R_trans; I(bl5,cell52) <+ V(bl5,cell52) /
      R_trans;
  I(bl6,cell62) <+ V(bl6,cell62) / R_trans; I(bl7,cell72) <+ V(bl7,cell72) /
      R_trans;
end
else begin
```

```
  I(bl0,cell02) <+ 0.0; I(bl1,cell12) <+ 0.0; I(bl2,cell22) <+ 0.0; I(bl3,cell32
      ) <+ 0.0;
  I(bl4,cell42) <+ 0.0; I(bl5,cell52) <+ 0.0; I(bl6,cell62) <+ 0.0; I(bl7,cell72
      ) <+ 0.0;
end

if (V(wl3) > V_on) begin
  I(bl0,cell03) <+ V(bl0,cell03) / R_trans; I(bl1,cell13) <+ V(bl1,cell13) /
      R_trans;
  I(bl2,cell23) <+ V(bl2,cell23) / R_trans; I(bl3,cell33) <+ V(bl3,cell33) /
      R_trans;
  I(bl4,cell43) <+ V(bl4,cell43) / R_trans; I(bl5,cell53) <+ V(bl5,cell53) /
      R_trans;
  I(bl6,cell63) <+ V(bl6,cell63) / R_trans; I(bl7,cell73) <+ V(bl7,cell73) /
      R_trans;
end
else begin
  I(bl0,cell03) <+ 0.0; I(bl1,cell13) <+ 0.0; I(bl2,cell23) <+ 0.0; I(bl3,cell33
      ) <+ 0.0;
  I(bl4,cell43) <+ 0.0; I(bl5,cell53) <+ 0.0; I(bl6,cell63) <+ 0.0; I(bl7,cell73
      ) <+ 0.0;
end

if (V(wl4) > V_on) begin
  I(bl0,cell04) <+ V(bl0,cell04) / R_trans; I(bl1,cell14) <+ V(bl1,cell14) /
      R_trans;
  I(bl2,cell24) <+ V(bl2,cell24) / R_trans; I(bl3,cell34) <+ V(bl3,cell34) /
      R_trans;
  I(bl4,cell44) <+ V(bl4,cell44) / R_trans; I(bl5,cell54) <+ V(bl5,cell54) /
      R_trans;
  I(bl6,cell64) <+ V(bl6,cell64) / R_trans; I(bl7,cell74) <+ V(bl7,cell74) /
      R_trans;
end
else begin
  I(bl0,cell04) <+ 0.0; I(bl1,cell14) <+ 0.0; I(bl2,cell24) <+ 0.0; I(bl3,cell34
      ) <+ 0.0;
  I(bl4,cell44) <+ 0.0; I(bl5,cell54) <+ 0.0; I(bl6,cell64) <+ 0.0; I(bl7,cell74
      ) <+ 0.0;
end

if (V(wl5) > V_on) begin
  I(bl0,cell05) <+ V(bl0,cell05) / R_trans; I(bl1,cell15) <+ V(bl1,cell15) /
      R_trans;
  I(bl2,cell25) <+ V(bl2,cell25) / R_trans; I(bl3,cell35) <+ V(bl3,cell35) /
      R_trans;
  I(bl4,cell45) <+ V(bl4,cell45) / R_trans; I(bl5,cell55) <+ V(bl5,cell55) /
      R_trans;
  I(bl6,cell65) <+ V(bl6,cell65) / R_trans; I(bl7,cell75) <+ V(bl7,cell75) /
      R_trans;
end
else begin
  I(bl0,cell05) <+ 0.0; I(bl1,cell15) <+ 0.0; I(bl2,cell25) <+ 0.0; I(bl3,cell35
      ) <+ 0.0;
  I(bl4,cell45) <+ 0.0; I(bl5,cell55) <+ 0.0; I(bl6,cell65) <+ 0.0; I(bl7,cell75
      ) <+ 0.0;
end

if (V(wl6) > V_on) begin
  I(bl0,cell06) <+ V(bl0,cell06) / R_trans; I(bl1,cell16) <+ V(bl1,cell16) /
      R_trans;
  I(bl2,cell26) <+ V(bl2,cell26) / R_trans; I(bl3,cell36) <+ V(bl3,cell36) /
      R_trans;
  I(bl4,cell46) <+ V(bl4,cell46) / R_trans; I(bl5,cell56) <+ V(bl5,cell56) /
      R_trans;
  I(bl6,cell66) <+ V(bl6,cell66) / R_trans; I(bl7,cell76) <+ V(bl7,cell76) /
      R_trans;
  end
  else begin
    I(bl0 , cell06 )  <+ 0.0; I(bl1 , cell16 )  <+ 0.0; I(bl2 , cell26 )  <+ 0.0; I(bl3 , cell36
        ) <+ 0.0;
    I(bl4 , cell46 )  <+ 0.0; I(bl5 , cell56 )  <+ 0.0; I(bl6 , cell66 )  <+ 0.0; I(bl7 , cell76
        ) <+ 0.0;
  end

  if (V(wl7) > V_on) begin
    I(bl0 , cell07 ) <+ V(bl0 , cell07 ) / R_trans ; I(bl1 , cell17 ) <+ V(bl1 , cell17 ) /
        R_trans ;
    I(bl2 , cell27 ) <+ V(bl2 , cell27 ) / R_trans ; I(bl3 , cell37 ) <+ V(bl3 , cell37 ) /
        R_trans ;
    I(bl4 , cell47 ) <+ V(bl4 , cell47 ) / R_trans ; I(bl5 , cell57 ) <+ V(bl5 , cell57 ) /
        R_trans ;
    I(bl6 , cell67 ) <+ V(bl6 , cell67 ) / R_trans ; I(bl7 , cell77 ) <+ V(bl7 , cell77 ) /
        R_trans ;
  end
  else begin
    I(bl0 , cell07 )  <+ 0.0; I(bl1 , cell17 )  <+ 0.0; I(bl2 , cell27 )  <+ 0.0; I(bl3 , cell37
        ) <+ 0.0;
    I(bl4 , cell47 )  <+ 0.0; I(bl5 , cell57 )  <+ 0.0; I(bl6 , cell67 )  <+ 0.0; I(bl7 , cell77
        ) <+ 0.0;
  end


  // Install bl conductance to aid Spectre
  I(bl0) <+ V(bl0) * G_bl;
  I(bl1) <+ V(bl1) * G_bl;
  I(bl2) <+ V(bl2) * G_bl;
  I(bl3) <+ V(bl3) * G_bl;
  I(bl4) <+ V(bl4) * G_bl;
  I(bl5) <+ V(bl5) * G_bl;
  I(bl6) <+ V(bl6) * G_bl;
  I(bl7) <+ V(bl7) * G_bl;

end


endmodule
```

Listing E.6: Multilevel Data Converter Code

```
// VerilogA for func_adc

`include "discipline.h"
`include "constants.h"

module func_adc(sr0 , sr1 , sr2 , sr3 , en , vin , dout0 , dout1);
input sr0 , sr1 , sr2 , sr3 , en , vin;
output dout0 , dout1;
electrical sr0 , sr1 , sr2 , sr3 , en , vin , dout0 , dout1;

parameter real V_th = 0.45;
parameter real Vdd = 1.2;

real ref0 , ref1 , ref2 , ref3;
real dout0_val , dout1_val;
real dist0 , dist1 , dist2 , dist3;

  analog begin
    @( initial_step ) begin
      V(dout0) <+ 0.0;
      I(dout0) <+ 0.0;
      V(dout1) <+ 0.0;
      I(dout1) <+ 0.0;
      ref0 = 0.0;
      ref1 = 0.0;
      ref2 = 0.0;
      ref3 = 0.0;
      dout0_val = 0.0;
      dout1_val = 0.0;
    end


    if ((V(sr0) > V_th) && (V(sr1) < V_th) && (V(sr2) < V_th) && (V(sr3) < V_th))
        begin
      ref0 = V(vin);
    end
    else if ((V(sr0) < V_th) && (V(sr1) > V_th) && (V(sr2) < V_th) && (V(sr3) <
      V_th)) begin
      ref1 = V(vin);
    end
    else if ((V(sr0) < V_th) && (V(sr1) < V_th) && (V(sr2) > V_th) && (V(sr3) <
        V_th)) begin
      ref2 = V(vin);
    end
    else if ((V(sr0) < V_th) && (V(sr1) < V_th) && (V(sr2) < V_th) && (V(sr3) >
        V_th)) begin
      ref3 = V(vin);
    end
    else if ((V(sr0) < V_th) && (V(sr1) < V_th) && (V(sr2) < V_th) && (V(sr3) <
        V_th)) begin
      // Calculate distances
      dist0 = abs(V(vin) - ref0);
      dist1 = abs(V(vin) - ref1);
      dist2 = abs(V(vin) - ref2);
      dist3 = abs(V(vin) - ref3);

      if ((dist0 < dist1) && (dist0 < dist2) && (dist0 < dist3)) begin
        dout0_val = 0.0;
        dout1_val = 0.0;
      end
      else if ((dist1 < dist0) && (dist1 < dist2) && (dist1 < dist3)) begin
        dout0_val = Vdd;
        dout1_val = 0.0;
      end
      else if ((dist2 < dist0) && (dist2 < dist1) && (dist2 < dist3)) begin
        dout0_val = 0.0;
        dout1_val = Vdd;
      end
      else begin
        dout0_val = Vdd;
        dout1_val = Vdd;
      end
    end

    if (V(en) > V_th) begin
      V(dout0) <+ dout0_val;
      V(dout1) <+ dout1_val;
    end
    else begin
      V(dout0) <+ 0.0;
      V(dout1) <+ 0.0;
    end

  end

endmodule
```

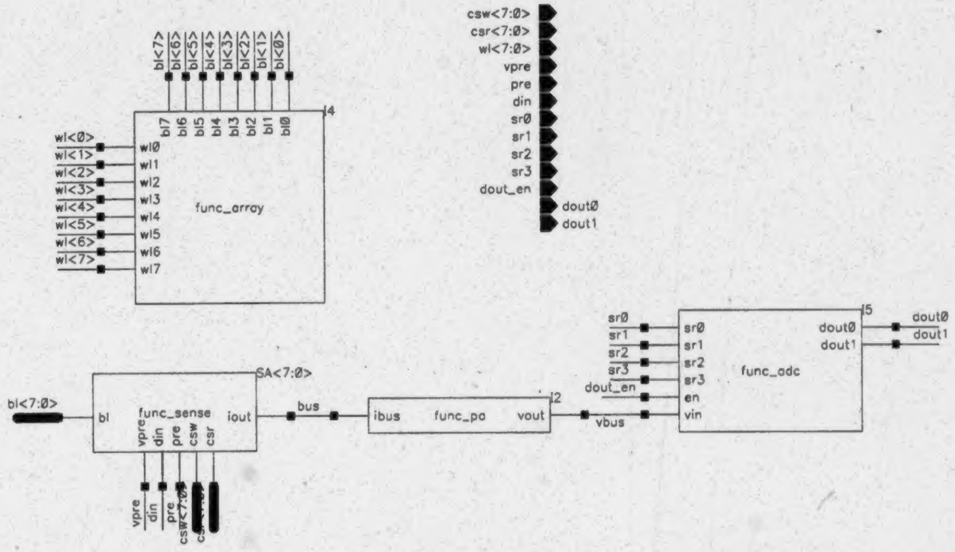

func_array
I4
wl<0>
wl<1>
wl<2>
wl<3>
wl<4>
wl<5>
wl<6>
wl<7>
wl0
wl1
wl2
wl3
wl4
wl5
wl6
wl7
bl7
bl6
bl5
bl4
bl3
bl2
bl1
bl0
bl<7>
bl<6>
bl<5>
bl<4>
bl<3>
bl<2>
bl<1>
bl<0>
csw<7:0>
csr<7:0>
wl<7:0>
vpre
pre
din
sr0
sr1
sr2
sr3
dout_en
dout0
dout1
func_sense
SA<7:0>
bl<7:0>
bl
vpre
din
pre
csw
csr
iout
vpre
din
pre
csw<7:0>
bus
func_pa
I2
ibus
vout
vbus
func_adc
I5
sr0
sr1
sr2
sr3
dout_en
sr0
sr1
sr2
sr3
en
vin
dout0
dout1
dout0
dout1

Figure E.2: Top Level Interconnection Schematic for Multilevel Functional Simulation